Projekt

On-line obchod s DVD-čkami. Program obsahuje katalog Dvdečiek. Taktiež rôzne druhy dvdečiek, ktoré obsahuje trieda DVD. Zákazníci si môžu objednať DVDčka. Hodnoty sú načítavané z databázy nove\_dvd. Taktiež obsahuje aj evidenciu používateľov. Ich mená a hesla. Evidujú sa objednávky užívateľov. Môžeme do databázy vkladať nové druhy DVDčiek a taktiež vymazať tie, ktoré sa už nepredávajú. Zoznam nových DVD, vymazať z on-line stránky stare DVD, pridávať nové DVD na predaj a upravovať zoznam DVD.

Triedy kde su infomácie o DVD, Použivateloch, Katalog

package dvd.data;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import javax.sql.DataSource;

public class DataAccessObject

{

private static DataSource dataSource;

private static Object idLock = new Object();

public static void setDataSource(DataSource dataSource)

{

DataAccessObject.dataSource = dataSource;

}

protected static Connection getConnection()

{

try

{

return dataSource.getConnection();

} catch (SQLException e)

{

throw new RuntimeException(e);

}

}

protected static void close(Statement statement, Connection connection)

{

close(null, statement, connection);

}

protected static void close(ResultSet rs, Statement statement,

Connection connection)

{

try

{

if (rs != null)

rs.close();

if (statement != null)

statement.close();

if (connection != null)

connection.close();

} catch (SQLException e)

{

throw new RuntimeException(e);

}

}

protected static Long getUniqueId()

{

ResultSet rs = null;

PreparedStatement statement = null;

Connection connection = null;

try

{

connection = getConnection();

synchronized (idLock)

{

statement = connection

.prepareStatement("select next\_value from objednanie");

rs = statement.executeQuery();

rs.first();

long id = rs.getLong(1);

statement.close();

statement = connection

.prepareStatement("update objednanie set next\_value = ?");

statement.setLong(1, id + 1);

statement.executeUpdate();

statement.close();

return new Long(id);

}

} catch (SQLException e)

{

throw new RuntimeException(e);

} finally

{

close(rs, statement, connection);

}

}

}

**package** dvd.data;

**public** **class** DVD

{

**private** String itemID;

**private** String popis;

**private** **double** cena;

**public** DVD(String itemID, String popis, **double** cena) {

setItemID(itemID);

setPopis(popis);

setCena(cena);

}

**public** String getItemID() {

**return** (itemID);

}

**protected** **void** setItemID(String itemID) {

**this**.itemID = itemID;}

**public** String getPopis()

{

**return**(popis);

}

**protected** **void** setPopis(String popis) {

**this**.popis = popis; }

**public** **double** getCena() {

**return** (cena);}

**protected** **void** setCena(**double** cena) {

**this**.cena = cena;

}

}

**package** dvd.data;

**public** **class** DVDOrder

{

**private** DVD item;

**private** **int** PocetItems;

**public** DVDOrder(DVD item) {

setItem(item);

setPocetItems(1);

}

**public** DVD getItem() {

**return** (item); }

**protected** **void** setItem(DVD item ) {

**this**.item = item;

}

**public** String getItemID(){

**return**(getItem().getItemID());}

**public** String getPopis() {

**return**(getItem().getPopis());

}

**public** **double** getUnitCena(){

**return**(getItem().getCena()); }

**public** **int** getPocetItems() {

**return**(PocetItems);}

**public** **void** setPocetItems(**int** n){

**this**.PocetItems = n;}

**public** **void** incrementPocetItems() {

setPocetItems(getPocetItems() + 1); }

**public** **void** cancelOrder() {

setPocetItems(0);}

**public** **double** getTotalCena() {

**return**(getPocetItems() \* getUnitCena());

}

}

package dvd.data;

import java.io.IOException;

import java.net.URL;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletConfig;

import javax.servlet.ServletContext;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.apache.log4j.Logger;

import com.sun.syndication.feed.synd.SyndFeed;

import com.sun.syndication.io.FeedException;

import com.sun.syndication.io.SyndFeedInput;

import com.sun.syndication.io.XmlReader;

public class HomeServlet extends HttpServlet

{

private Logger logger = Logger.getLogger(this.getClass());

private RequestDispatcher homeJsp;

@Override

public void init(ServletConfig config) throws ServletException

{

ServletContext context = config.getServletContext();

homeJsp = context.getRequestDispatcher("/web/WEB-INF/home.jsp");

}

@Override

protected void doGet(HttpServletRequest req, HttpServletResponse resp)

throws ServletException, IOException

{

logger.debug("Odpoved");

URL url = new URL("http://rss.news.yahoo.com/rss/tech");

SyndFeedInput syndFeedInput = new SyndFeedInput();

SyndFeed syndFeed = null;

XmlReader xmlReader = new XmlReader(url);

try

{

syndFeed = syndFeedInput.build(xmlReader);

} catch (IllegalArgumentException e)

{

logger.error("", e);

} catch (FeedException e)

{

logger.error("", e);

}

logger.debug("Vratit sa spet na home.jsp");

req.setAttribute("syndFeed", syndFeed);

homeJsp.forward(req, resp);

}

}

**package** dvd.data;

**public** **class** Katalog

{

**private** **static** DVD[] *items*;

**public** **static** DVD getItem(String itemID) {

DVD item;

**if**(itemID == **null**) {

**return**(**null**); }

**for**(**int** i=0; i<*items*.length; i++){

item = *items*(i);

**if**(itemID.equals(item.getItemID())) {

**return**(item);

}

}

**return** (**null**);

}

**private** **static** DVD items(**int** i)

{

**return** **null**;

}

}

package dvd.data;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.LinkedList;

import java.util.List;

public class NoveDVD\_DAO extends DataAccessObject

{

private NoveDVD read(ResultSet rs) throws SQLException

{

Long id = new Long(rs.getLong("id"));

String titul = rs.getString("titul");

String nazov = rs.getString("nazov");

NoveDVD newsItem = new NoveDVD();

newsItem.setId(id);

newsItem.setTitul(titul);

newsItem.setNazov(nazov);

return newsItem;

}

public List<NoveDVD> findAll()

{

LinkedList<NoveDVD> newsItems = new LinkedList<NoveDVD>();

ResultSet rs = null;

PreparedStatement statement = null;

Connection connection = null;

try

{

connection = getConnection();

String sql = "select \* from nove\_dvd order by id";

statement = connection.prepareStatement(sql);

rs = statement.executeQuery();

while (rs.next())

{

NoveDVD newsItem = read(rs);

newsItems.add(newsItem);

}

return newsItems;

} catch (SQLException e)

{

throw new RuntimeException(e);

} finally

{

close(rs, statement, connection);

}

}

public NoveDVD find(Long id)

{

ResultSet rs = null;

PreparedStatement statement = null;

Connection connection = null;

try

{

connection = getConnection();

String sql = "select \* from nove\_dvd where id=?";

statement = connection.prepareStatement(sql);

statement.setLong(1, id.longValue());

rs = statement.executeQuery();

if (!rs.next())

{

return null;

}

return read(rs);

} catch (SQLException e)

{

throw new RuntimeException(e);

} finally

{

close(rs, statement, connection);

}

}

public void update(NoveDVD newsItem)

{

PreparedStatement statement = null;

Connection connection = null;

try

{

connection = getConnection();

String sql = "update nove\_dvd set " + "titul=?, nazov=? where id=?";

statement = connection.prepareStatement(sql);

statement.setString(1, newsItem.getTitul());

statement.setString(2, newsItem.getNazov());

statement.setLong(3, newsItem.getId().longValue());

statement.execute();

} catch (SQLException e)

{

throw new RuntimeException(e);

} finally

{

close(statement, connection);

}

}

public void create(NoveDVD newsItem)

{

Long id = getUniqueId();

newsItem.setId(id);

PreparedStatement statement = null;

Connection connection = null;

try

{

connection = getConnection();

String sql = "insert into nove\_dvd " + "(id, titul, nazov) "

+ "values (?, ?, ?)";

statement = connection.prepareStatement(sql);

statement.setLong(1, id.longValue());

statement.setString(2, newsItem.getTitul());

statement.setString(3, newsItem.getNazov());

statement.executeUpdate();

} catch (SQLException e)

{

throw new RuntimeException(e);

} finally

{

close(statement, connection);

}

}

public void delete(NoveDVD newsItem)

{

PreparedStatement statement = null;

Connection connection = null;

try

{

connection = getConnection();

String sql = "delete from nove\_dvd where id=?";

statement = connection.prepareStatement(sql);

Long id = newsItem.getId();

statement.setLong(1, id.longValue());

statement.executeUpdate();

} catch (SQLException e)

{

throw new RuntimeException(e);

} finally

{

close(statement, connection);

}

}

}

**package** dvd.data;

**public** **class** NoveDVD

{

**private** Long id;

**private** String titul;

**private** String nazov;

**public** Long getId()

{

**return** id;

}

**public** **void** setId(Long id)

{

**this**.id = id;

}

**public** String getTitul()

{

**return** titul;

}

**public** **void** setTitul(String titul)

{

**this**.titul = titul;

}

**public** String getNazov()

{

**return** nazov;

}

**public** **void** setNazov(String nazov)

{

**this**.nazov = nazov;

}

}

**package** dvd.data;

**import** java.util.Vector;

**public** **class** ShoppingCart

{

**private** Vector<DVDOrder> itemsOrdered;

**public** ShoppingCart() {

itemsOrdered = **new** Vector<DVDOrder> ();

}

**public** Vector<DVDOrder> getItemsOrdered (){

**return**(itemsOrdered);

}

**public** **synchronized** **void** addItem(String itemID) {

DVDOrder order;

**for**(**int** i=0; i<itemsOrdered.size(); i++) {

order = itemsOrdered.elementAt(i);

**if**(order.getItemID().equals(itemID)) {

order.incrementPocetItems();

**return**;

}

}

DVDOrder newOrder = **new** DVDOrder (Katalog.*getItem*(itemID));

itemsOrdered.addElement(newOrder);

}

**public** **synchronized** **void** setPocetOrdered(String itemID, **int** numOrdered) {

DVDOrder order;

**for**(**int** i=0; i<itemsOrdered.size(); i++) {

order = itemsOrdered.elementAt(i);

**if** (order.getItemID().equals(itemID)) {

**if**(numOrdered <= 0) {

itemsOrdered.removeElementAt(i);

} **else** {

order.setPocetItems(numOrdered);

}

**return**;

}

}

DVDOrder newOrder = **new** DVDOrder(Katalog.*getItem*(itemID));

itemsOrdered.addElement(newOrder);

}

}

**package** dvd.data;

**public** **class** User

{

**private** Long id;

**private** String username;

**private** String password;

**public** Long getId()

{

**return** id;

}

**public** **void** setId(Long id)

{

**this**.id = id;

}

**public** String getPassword()

{

**return** password;

}

**public** **void** setPassword(String password)

{

**this**.password = password;

}

**public** String getUsername()

{

**return** username;

}

**public** **void** setUsername(String username)

{

**this**.username = username;

}

}

package dvd.data;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.LinkedList;

import java.util.List;

public class UserDAO extends DataAccessObject

{

private static UserDAO instance = new UserDAO();

public static UserDAO getInstance()

{

return instance;

}

private User read(ResultSet rs) throws SQLException

{

Long id = new Long(rs.getLong("id"));

String username = rs.getString("username");

String password = rs.getString("password");

User user = new User();

user.setId(id);

user.setUsername(username);

user.setPassword(password);

return user;

}

public User find(Long id)

{

ResultSet rs = null;

PreparedStatement statement = null;

Connection connection = null;

try

{

connection = getConnection();

String sql = "select \* from user where id=?";

statement = connection.prepareStatement(sql);

statement.setLong(1, id.longValue());

rs = statement.executeQuery();

if (!rs.next())

{

return null;

}

return read(rs);

} catch (SQLException e)

{

throw new RuntimeException(e);

} finally

{

close(rs, statement, connection);

}

}

public User findByUsername(String username)

{

ResultSet rs = null;

PreparedStatement statement = null;

Connection connection = null;

try

{

connection = getConnection();

String sql = "select \* from user where username=?";

statement = connection.prepareStatement(sql);

statement.setString(1, username);

rs = statement.executeQuery();

if (!rs.next())

{

return null;

}

return read(rs);

} catch (SQLException e)

{

throw new RuntimeException(e);

} finally

{

close(rs, statement, connection);

}

}

public List<User> findAll()

{

LinkedList<User> users = new LinkedList<User>();

ResultSet rs = null;

PreparedStatement statement = null;

Connection connection = null;

try

{

connection = getConnection();

String sql = "select \* from user order by id";

statement = connection.prepareStatement(sql);

rs = statement.executeQuery();

while (rs.next())

{

User user = read(rs);

users.add(user);

}

return users;

} catch (SQLException e)

{

throw new RuntimeException(e);

} finally

{

close(rs, statement, connection);

}

}

public void update(User user)

{

PreparedStatement statement = null;

Connection connection = null;

try

{

connection = getConnection();

String sql = "update user set " + "password=? where id=?";

statement = connection.prepareStatement(sql);

statement.setString(1, user.getPassword());

statement.setLong(2, user.getId().longValue());

statement.executeUpdate();

} catch (SQLException e)

{

throw new RuntimeException(e);

} finally

{

close(statement, connection);

}

}

public void create(User user)

{

Long id = getUniqueId();

user.setId(id);

PreparedStatement statement = null;

Connection connection = null;

try

{

connection = getConnection();

String sql = "insert into user " + "(id, username, password) "

+ "values (?, ?, ?)";

statement = connection.prepareStatement(sql);

statement.setLong(1, id.longValue());

statement.setString(2, user.getUsername());

statement.setString(3, user.getPassword());

statement.executeUpdate();

} catch (SQLException e)

{

throw new RuntimeException(e);

} finally

{

close(statement, connection);

}

}

public void delete(User user)

{

PreparedStatement statement = null;

Connection connection = null;

try

{

connection = getConnection();

String sql = "delete from user where id=?";

statement = connection.prepareStatement(sql);

Long id = user.getId();

statement.setLong(1, id.longValue());

statement.executeUpdate();

} catch (SQLException e)

{

throw new RuntimeException(e);

} finally

{

close(statement, connection);

}

}

}

Triedy so Servletmi:

package dvd.data;

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.LinkedList;

import java.util.List;

public class UserDAO extends DataAccessObject

{

private static UserDAO instance = new UserDAO();

public static UserDAO getInstance()

{

return instance;

}

private User read(ResultSet rs) throws SQLException

{

Long id = new Long(rs.getLong("id"));

String username = rs.getString("username");

String password = rs.getString("password");

User user = new User();

user.setId(id);

user.setUsername(username);

user.setPassword(password);

return user;

}

public User find(Long id)

{

ResultSet rs = null;

PreparedStatement statement = null;

Connection connection = null;

try

{

connection = getConnection();

String sql = "select \* from user where id=?";

statement = connection.prepareStatement(sql);

statement.setLong(1, id.longValue());

rs = statement.executeQuery();

if (!rs.next())

{

return null;

}

return read(rs);

} catch (SQLException e)

{

throw new RuntimeException(e);

} finally

{

close(rs, statement, connection);

}

}

public User findByUsername(String username)

{

ResultSet rs = null;

PreparedStatement statement = null;

Connection connection = null;

try

{

connection = getConnection();

String sql = "select \* from user where username=?";

statement = connection.prepareStatement(sql);

statement.setString(1, username);

rs = statement.executeQuery();

if (!rs.next())

{

return null;

}

return read(rs);

} catch (SQLException e)

{

throw new RuntimeException(e);

} finally

{

close(rs, statement, connection);

}

}

public List<User> findAll()

{

LinkedList<User> users = new LinkedList<User>();

ResultSet rs = null;

PreparedStatement statement = null;

Connection connection = null;

try

{

connection = getConnection();

String sql = "select \* from user order by id";

statement = connection.prepareStatement(sql);

rs = statement.executeQuery();

while (rs.next())

{

User user = read(rs);

users.add(user);

}

return users;

} catch (SQLException e)

{

throw new RuntimeException(e);

} finally

{

close(rs, statement, connection);

}

}

public void update(User user)

{

PreparedStatement statement = null;

Connection connection = null;

try

{

connection = getConnection();

String sql = "update user set " + "password=? where id=?";

statement = connection.prepareStatement(sql);

statement.setString(1, user.getPassword());

statement.setLong(2, user.getId().longValue());

statement.executeUpdate();

} catch (SQLException e)

{

throw new RuntimeException(e);

} finally

{

close(statement, connection);

}

}

public void create(User user)

{

Long id = getUniqueId();

user.setId(id);

PreparedStatement statement = null;

Connection connection = null;

try

{

connection = getConnection();

String sql = "insert into user " + "(id, username, password) "

+ "values (?, ?, ?)";

statement = connection.prepareStatement(sql);

statement.setLong(1, id.longValue());

statement.setString(2, user.getUsername());

statement.setString(3, user.getPassword());

statement.executeUpdate();

} catch (SQLException e)

{

throw new RuntimeException(e);

} finally

{

close(statement, connection);

}

}

public void delete(User user)

{

PreparedStatement statement = null;

Connection connection = null;

try

{

connection = getConnection();

String sql = "delete from user where id=?";

statement = connection.prepareStatement(sql);

Long id = user.getId();

statement.setLong(1, id.longValue());

statement.executeUpdate();

} catch (SQLException e)

{

throw new RuntimeException(e);

} finally

{

close(statement, connection);

}

}

}

package dvd.online.objednaj.web;

import javax.naming.Context;

import javax.naming.InitialContext;

import javax.servlet.ServletContext;

import javax.servlet.ServletContextEvent;

import javax.servlet.ServletContextListener;

import javax.sql.DataSource;

import org.apache.log4j.Logger;

import dvd.data.DataAccessObject;

public class Init implements ServletContextListener

{

private Logger logger = Logger.getLogger(this.getClass());

@Override

public void contextDestroyed(ServletContextEvent sce)

{

}

private void contextInitialized2(ServletContext servletContext)

throws Exception

{

InitialContext enc = new InitialContext();

Context compContext = (Context) enc.lookup("java:comp/env");

DataSource dataSource = (DataSource) compContext.lookup("datasource");

DataAccessObject.setDataSource(dataSource);

}

@Override

public void contextInitialized(ServletContextEvent sce)

{

ServletContext servletContext = sce.getServletContext();

try

{

contextInitialized2(servletContext);

} catch (Exception e)

{

logger.error("Inicializácia sa nepodarila", e);

throw new RuntimeException(e);

}

logger.debug("Inicializácia je úspešná :).");

}

}

package dvd.online.objednaj.web;

import java.io.IOException;

import java.io.Writer;

import java.util.ArrayList;

import java.util.Iterator;

import java.util.List;

import javax.servlet.ServletConfig;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.apache.log4j.Logger;

import com.sun.syndication.feed.synd.SyndEntry;

import com.sun.syndication.feed.synd.SyndEntryImpl;

import com.sun.syndication.feed.synd.SyndFeed;

import com.sun.syndication.feed.synd.SyndFeedImpl;

import com.sun.syndication.io.FeedException;

import com.sun.syndication.io.SyndFeedOutput;

import dvd.data.NoveDVD;

import dvd.data.NoveDVD\_DAO;

public class ObjServlet extends HttpServlet

{

private Logger logger = Logger.getLogger(this.getClass());

@Override

public void init(ServletConfig config) throws ServletException

{

logger.debug("init()");

try

{

Class.forName("com.mysql.jdbc.Driver");

} catch (ClassNotFoundException e)

{

throw new ServletException(e);

}

}

@Override

protected void doGet(HttpServletRequest req, HttpServletResponse resp)

throws ServletException, IOException

{

SyndFeed feed = new SyndFeedImpl();

feed.setFeedType("rss\_2.0");

feed.setTitle("Harry");

feed.setLink("http://localhost:8080/MARGETOVA\_PROJEKT\_PRVA\_VERZIA/");

feed.setDescription("Tu su objednane");

List<SyndEntry> entries = new ArrayList<SyndEntry>();

List<NoveDVD> newsItems = new NoveDVD\_DAO().findAll();

Iterator<NoveDVD> it = newsItems.iterator();

while (it.hasNext())

{

NoveDVD newsItem = (NoveDVD) it.next();

String titul = newsItem.getTitul();

String nazov = newsItem.getNazov();

SyndEntry entry = new SyndEntryImpl();

entry.setTitle(titul);

entry.setLink(nazov);

entries.add(entry);

resp.setContentType("text/xml");

feed.setEntries(entries);

Writer writer = resp.getWriter();

SyndFeedOutput output = new SyndFeedOutput();

try

{

output.output(feed, writer);

} catch (FeedException e)

{

logger.error("", e);

}

}

}

}

package dvd.online.objednaj.web;

import java.io.IOException;

import java.util.Map;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletConfig;

import javax.servlet.ServletContext;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.apache.log4j.Logger;

import dvd.data.NoveDVD;

import dvd.data.NoveDVD\_DAO;

public class PridajNoveDVDServlet extends HttpServlet

{

private Logger logger = Logger.getLogger(this.getClass());

private RequestDispatcher jsp;

public void init(ServletConfig config) throws ServletException

{

ServletContext context = config.getServletContext();

jsp = context.getRequestDispatcher("/web/WEB-INF/jsp/uprav-nove-dvd.jsp");

}

protected void doGet(HttpServletRequest req, HttpServletResponse resp)

throws ServletException, IOException

{

logger.debug("doGet()");

jsp.forward(req, resp);

}

protected void doPost(HttpServletRequest req, HttpServletResponse resp)

throws ServletException, IOException

{

String cancelButton = req.getParameter("zrusenie - tlacitko");

if (cancelButton != null)

{

logger.debug("Zrusenie");

resp.sendRedirect("zoznam-nove-dvd");

return;

}

Map<String, String> errors = UpravNoveDVDServlet.validate(req);

if (!errors.isEmpty())

{

logger.debug("Chyba");

jsp.forward(req, resp);

return;

}

NoveDVD newsItem = (NoveDVD) req.getAttribute("newsItem");

new NoveDVD\_DAO().create(newsItem);

resp.sendRedirect("uloz-nove-dvd?id=" + newsItem.getId());

}

}

package dvd.online.objednaj.web;

import java.io.IOException;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletConfig;

import javax.servlet.ServletContext;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.apache.log4j.Logger;

import dvd.data.NoveDVD;

import dvd.data.NoveDVD\_DAO;

public class UlozNoveDVDServlet extends HttpServlet

{

private Logger logger = Logger.getLogger(this.getClass());

private RequestDispatcher jsp;

public void init(ServletConfig config) throws ServletException

{

ServletContext context = config.getServletContext();

jsp = context.getRequestDispatcher("/web/WEB-INF/jsp/uloz-nove-dvd.jsp");

}

protected void doGet(HttpServletRequest req, HttpServletResponse resp)

throws ServletException, IOException

{

logger.debug("doGet()");

String idString = req.getParameter("id");

Long id = new Long(idString);

NoveDVD newsItem = new NoveDVD\_DAO().find(id);

req.setAttribute("newsItem", newsItem);

jsp.forward(req, resp);

}

}

package dvd.online.objednaj.web;

import java.io.IOException;

import java.util.HashMap;

import java.util.Map;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletConfig;

import javax.servlet.ServletContext;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.apache.log4j.Logger;

import dvd.data.NoveDVD;

import dvd.data.NoveDVD\_DAO;

public class UpravNoveDVDServlet extends HttpServlet

{

private Logger logger = Logger.getLogger(this.getClass());

private RequestDispatcher jsp;

public void init(ServletConfig config) throws ServletException

{

ServletContext context = config.getServletContext();

jsp = context.getRequestDispatcher("/web/WEB-INF/jsp/uloz-nove-dvd.jsp");

}

protected void doGet(HttpServletRequest req, HttpServletResponse resp)

throws ServletException, IOException

{

logger.debug("doGet()");

String idString = req.getParameter("id");

Long id = new Long(idString);

NoveDVD newsItem = new NoveDVD\_DAO().find(id);

req.setAttribute("newsItem", newsItem);

jsp.forward(req, resp);

}

protected void doPost(HttpServletRequest req, HttpServletResponse resp)

throws ServletException, IOException

{

String id = req.getParameter("id");

String cancelButton = req.getParameter("zrusenie - tlacitko");

if (cancelButton != null)

{

logger.debug("zrusenie");

resp.sendRedirect("zoznam-nove-dvd?id=" + id);

return;

}

Map<String, String> errors = validate(req);

if (!errors.isEmpty())

{

logger.debug("chyba kontroly");

jsp.forward(req, resp);

return;

}

NoveDVD newsItem = (NoveDVD) req.getAttribute("newsItem");

new NoveDVD\_DAO().update(newsItem);

resp.sendRedirect("uloz-nove-dvd?id=" + id);

}

public static Map<String, String> validate(HttpServletRequest req)

{

NoveDVD newsItem = new NoveDVD();

HashMap<String, String> errors = new HashMap<String, String>();

req.setAttribute("errors", errors);

req.setAttribute("newsItem", newsItem);

String idString = req.getParameter("id");

if (idString != null && idString.length() > 0)

{

Long id = new Long(idString);

newsItem.setId(id);

}

String titul = req.getParameter("titul");

if (titul == null || titul.trim().length() == 0)

{

errors.put("titul", "Titul .");

}

newsItem.setTitul(titul);

String nazov = req.getParameter("nazov");

if (nazov == null || nazov.trim().length() == 0)

{

errors.put("nazov", "nazov dvd");

}

newsItem.setNazov(nazov);

return errors;

}

}

package dvd.online.objednaj.web;

import java.io.IOException;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletConfig;

import javax.servlet.ServletContext;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.apache.log4j.Logger;

import dvd.data.NoveDVD;

import dvd.data.NoveDVD\_DAO;

public class VymazDVDServlet extends HttpServlet

{

private Logger logger = Logger.getLogger(this.getClass());

private RequestDispatcher jsp;

public void init(ServletConfig config) throws ServletException {

ServletContext context = config.getServletContext();

jsp = context.getRequestDispatcher("/web/WEB-INF/jsp/delete-news-item.jsp");

}

protected void doGet(HttpServletRequest req, HttpServletResponse resp)

throws ServletException, IOException

{

logger.debug("doGet()");

jsp.forward(req, resp);

}

protected void doPost(HttpServletRequest req, HttpServletResponse resp)

throws ServletException, IOException

{

String idString = req.getParameter("id");

String cancelButton = req.getParameter("cancel-button");

if (cancelButton != null)

{

logger.debug("cancel button pressed");

resp.sendRedirect("view-news-item?id=" + idString);

return;

}

NoveDVD\_DAO newsItemDAO = new NoveDVD\_DAO();

Long id = new Long(idString);

NoveDVD newsItem = newsItemDAO.find(id);

new NoveDVD\_DAO().delete(newsItem);

resp.sendRedirect("zoznam-nove-dvd");

}

}

package dvd.online.objednaj.web;

import java.io.IOException;

import java.util.List;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletConfig;

import javax.servlet.ServletContext;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.apache.log4j.Logger;

import dvd.data.NoveDVD;

import dvd.data.NoveDVD\_DAO;

public class ZoznamNoveDVDServlet extends HttpServlet

{

private Logger logger = Logger.getLogger(this.getClass());

private RequestDispatcher jsp;

public void init(ServletConfig config) throws ServletException

{

ServletContext context = config.getServletContext();

jsp = context.getRequestDispatcher("/web/WEB-INF/jsp/zoznam-nove-dvd.jsp");

}

protected void doGet(HttpServletRequest req, HttpServletResponse resp)

throws ServletException, IOException

{

logger.debug("doGet()");

List<NoveDVD> newsItems = new NoveDVD\_DAO().findAll();

req.setAttribute("newsItems", newsItems);

jsp.forward(req, resp);

}

}

package dvd.online.objednaj.web;

import java.io.IOException;

import java.io.PrintWriter;

import java.util.Vector;

import javax.servlet.ServletException;

import javax.servlet.ServletResponse;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;

import dvd.data.DVDOrder;

import dvd.data.ShoppingCart;

public class OrderFilm extends HttpServlet

{

@SuppressWarnings("unused")

private ServletResponse response;

@SuppressWarnings({ "deprecation", "unused" })

public void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

HttpSession session = request.getSession(true);

ShoppingCart cart;

synchronized (session) {

cart = (ShoppingCart)session.getValue("shoppingCart");

if (cart == null) {

cart = new ShoppingCart();

session.putValue("shoppingCart", cart);

}

String itemID = request.getParameter("itemID");

response.setContentType("text/html");

PrintWriter out = response.getWriter();

String titul = "HP";

out.println(Titul (titul) +

"<h1 align=\"center\">" + titul + "</h1>");

synchronized(session) {

@SuppressWarnings("rawtypes")

Vector itemsOrdered = cart.getItemsOrdered();

if (itemsOrdered.size() == 0) {

out.println("<H2><I>Ziadne polozky v tvojej karte </I></H2>");

} else {

out.println

("<table border=1 align=\"center\">\n" +

"<tr bgcolor=\"red\">\n" +

"<th>Item ID<th>Popis\n" +

"<th>Cena<th>Pocet<th>Celkova cena");

DVDOrder order;

String formURL = "/servlet/Online\_obchod.OrderPage"; } } } }

private String Titul(String titul)

{

return null;

}

public void doPost (HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

doGet (request, response);

}

}

Databaza:

<project name="MARGETOVA\_PROJEKT" default="all" basedir=".">

<property name="mysql.params" value="-u MARGETOVA\_PROJEKT -pMARGETOVA\_PROJEKT -D MARGETOVA\_PROJEKT" />

<target name="all" depends="cleandb, createdb, insertdb"></target>

<target name="cleandb">

<exec executable="mysql" input="cleandb.sql">

<arg line="${mysql.params}" />

</exec>

</target>

<target name="createdb">

<exec executable="mysql" input="createdb.sql">

<arg line="${mysql.params}" />

</exec>

</target>

<target name="insertdb">

<exec executable="mysql" input="insertdb.sql">

<arg line="${mysql.params}" />

</exec>

</target>

</project>

**drop** **table** if **exists** nove\_dvd;

**drop** **table** if **exists** objednanie;

**drop** **table** if **exists** **user**;

**create** **table** nove\_dvd

(

id **integer** **primary** **key**,

titul text **not** **null**,

nazov text **not** **null**

);

**create** **table** **user**

(

id **integer** **primary** **key**,

username **varchar**(255) **unique**,

password **varchar**(255)

);

**create** **table** objednanie

(

next\_value **integer**

);

**insert** **into** objednanie **value** (1);

**insert** **into** nove\_dvd (id, titul, nazov) **values** (1, 'HP', 'AA');

**insert** **into** nove\_dvd (id, titul, nazov) **values** (2, 'AE', 'iu');

**insert** **into** **user** (id, username, password) **values** (4, 'admin', 'admin');

**insert** **into** **user** (id, username, password) **values** (5, 'harry', 'harry');

**insert** **into** **user** (id, username, password) **values** (6, 'emma', 'emma');

HTML:

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

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns=*"http://www.w3.org/1999/xhtml"*>

<head>

<title>DVD online</title>

</head>

<body style="margin: *0*; padding: *0*">

<table width=*"728"*

cellspacing=*"0"*

cellpadding=*"0"*

style="padding: *0*; margin: *0*; border-collapse: *collapse*">

<tr style="padding: *0*; margin: *0*">

<td width=*"728"*

colspan=*"2"*

style="padding: *0*; margin: *0*"></td>

</tr>

<tr style="padding: *0*; margin: *0*">

<td width=*"728"*

colspan=*"2"*

style="padding: *0*; margin: *0*"><div

style="padding: *0*;

margin: *0*;

text-align: *center*;

font-size: *small*;

background-color: *#99CCCC*">

DVD

</div></td>

</tr>

<tr>

<td rowspan=*"2"*

width=*"240"*

valign=*"top"*>

<div style="height: *500px*; padding: *20px*; border-right: *3px dashed*; margin-right: *20px*">

<a href=*"zoznam-nove-dvd"*>Nase nove DVD</a><br/>

<a href=*"uprav-nove-dvd"*>DVD-čka</a><br/>

<a href=*"logout"*>Odhlasenie</a> <br/>

</div>

</td>

<td width=*"488"*>

<div style="margin-top: *12px*; margin-bottom: *12 px*">

<a href=*"edit?item=3"*>edit</a>

<a href=*"delete?item=3"*>delete</a>

</div>

</td>

</tr>

<tr>

<td valign=*"top"*>

<h1 style="font-size: *larger*">Zoznam DVD</h1>

<p>Titul: HP a kamen</p>

<p>Link: <a href=*"http://localhost:8080/MARGETOVA\_PROJEKT"*>A</a></p>

</td>

</tr>

</table>

</body>

</html>

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

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns=*"http://www.w3.org/1999/xhtml"*>

<head>

<meta http-equiv=*"Content-Type"* content=*"text/html; charset=UTF-8"* />

<title>Filmy</title>

</head>

<body>

Tu su filmy<br/>

<ul>

<li><a href=*"home.jsp"*>Film</a></li>

</ul>

</body>

</html>

JSP:

</td>

</tr>

</table>

</body>

</html

<%@ include file="top.inc" %>

<%@ include file="middle.inc" %>

<h1>Vitajte</h1>

<p>

Obchod s DVD

</p>

<%@ include file="bottom.inc" %>

<%@ include file="top.inc" %>

<%@ include file="middle.inc" %>

<h1>Vitajte</h1>

<p>

Obchod s DVD

</p>

<%@ include file="bottom.inc" %>

<html>

<head>

<title>Hlavna</title>

</head>

<body style="margin: 0; padding: 0">

<table width="728"

cellspacing="0"

cellpadding="0"

style="padding: 0; margin: 0; border-collapse: collapse">

<tr style="padding: 0; margin: 0">

<td width="728"

colspan="2"

style="padding: 0; margin: 0"><img

src="logo.gif"

style="padding: 0; margin"></td>

</tr>

<tr style="padding: 0; margin: 0">

<td width="728"

colspan="2"

style="padding: 0; margin: 0"><div

style="padding: 0;

margin: 0;

text-align: center;

font-size: small;

background-color: #99CCCC">

Hlavna

</div></td>

</tr>

<tr>

<td rowspan="2"

width="240"

valign="top">

<div style="height: 500px; padding: 20px; border-right: 3px dashed; margin-right: 20px">

<a href="zoznam-nove-dvd">Home</a><br/>

<a href="uloz-nove-dvd">Nove</a><br/>

<a href="logout">Koniec</a> <br/>

</div>

</td>

<td width="488">

<div style="margin-top: 12px; margin-bottom: 12 px">

<%@ include file="top.inc" %>

<a href="uprav-nove-dvd?id=${newsItem.id}">Uprav</a>

<a href="vymaz-nove-dvd?id=${newsItem.id}">.Vymaz</a>

<%@ include file="middle.inc" %>

<table>

<tr>

<td>Titul:</td>

<td>${newsItem.titul}</td>

</tr>

<tr>

<td>Nazov:</td>

<td>

${newsItem.nazov}

<a href="${newsItem.nazov}">Nazov</a>

</td>

</tr>

</table>

<%@ include file="bottom.inc" %>

<jsp:useBean id="errors" scope="request" type="java.util.Map" class="java.util.HashMap" />

<%@ include file="top.inc" %>

<%@ include file="middle.inc" %>

<form method="post">

<table>

<tr>

<td>Title</td>

<td><input type="text" name="titul" value="${newsItem.titul}" size="50" />

<%

if (errors.containsKey("titul")) {

out.println("<span class=\"error\">" + errors.get("titul") + "</span>");

}

%>

</td>

</tr>

<tr>

<td>Nazov</td>

<td><input type="text" name="nazov" value="${newsItem.nazov}" size="50" />

<%

if (errors.containsKey("nazov")) {

out.println("<span class=\"error\">" + errors.get("nazov") + "</span>");

}

%>

</td>

</tr>

<tr>

<td>

<input type="submit" name="submit-button" value="Submit" />

<input type="submit" name="zrusenie - tlacitko" value="Cancel" />

</td>

</tr>

</table>

<input type="hidden" name="id" value="${newsItem.id}" />

</form>

<%@ include file="bottom.inc" %>

<%@ include file="top.inc" %>

<%@ include file="middle.inc" %>

<p>

Vymazat uz nepredavane dvdčka

</p>

<form method="post">

<input type="submit" name="delete-button" value="Delete" />

<input type="submit" name="cancel-button" value="Cancel" />

<input type="hidden" name="id" value="${params['id']}" />

</form>

<%@ include file="bottom.inc" %>

<%@ page import="java.util.Iterator" %>

<%@ page import="dvd.data.NoveDVD" %>

<jsp:useBean id="newsItem" scope="request" type="java.util.List" />

<%@ include file="top.inc" %>

<a href="pridaj-nove-dvd">Nove DVD</a>

<%@ include file="middle.inc" %>

<ul>

<%

Iterator it = newsItems.iterator();

while (it.hasNext())

{

NewsItem newsItem = (NewsItem) it.next();

%>

<li>

<a href="view-news-item?id=<%=newsItem.getId()%>"><%=newsItem.getTitul()%></a>

</li>

<% } %>

</ul>

<%@ include file="bottom.inc" %>

XML:

<?xml version=*"1.0"*?>

<web-app

xmlns=*"http://java.sun.com/xml/ns/j2ee"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://java.sun.com/xml/ns/j2ee http://java.sun.com/xml/ns/j2ee/web-app\_2\_4.xsd"*

version=*"2.4"*>

<listener>

<listener-class>dvd.online.objednaj.web.Init</listener-class>

</listener>

<servlet>

<servlet-name>Obj</servlet-name>

<servlet-class>dvd.online.objednaj.web.ObjServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>news-rss</servlet-name>

<url-pattern>/news.rss</url-pattern>

</servlet-mapping>

<servlet>

<servlet-name>home</servlet-name>

<servlet-class>dvd.online.objednaj.web.HomeServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>home</servlet-name>

<url-pattern>/home</url-pattern>

</servlet-mapping>

<servlet>

<servlet-name>zoznam-nove-dvd</servlet-name>

<servlet-class>dvd.online.objednaj.web.ZoznamNoveDVDServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>zoznam-nove-dvd</servlet-name>

<url-pattern>/zoznam-nove-dvd</url-pattern>

</servlet-mapping>

<servlet>

<servlet-name>uloz-nove-dvd</servlet-name>

<servlet-class>dvd.online.objednaj.web.UlozNoveDVDServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>uloz-nove-dvd</servlet-name>

<url-pattern>/uloz-nove-dvd</url-pattern>

</servlet-mapping>

<servlet>

<servlet-name>pridaj-nove-dvd</servlet-name>

<servlet-class>dvd.online.objednaj.web.PridajNoveDVDServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>pridaj-nove-dvd</servlet-name>

<url-pattern>/pridaj-nove-dvd</url-pattern>

</servlet-mapping>

<servlet>

<servlet-name>vymaz-stare-dvd</servlet-name>

<servlet-class>dvd.online.objednaj.web.VymazDVDServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>vymaz-stare-dvd</servlet-name>

<url-pattern>/vymaz-stare-dvd</url-pattern>

</servlet-mapping>

<resource-ref>

<description>dataSource</description>

<res-ref-name>dataSource</res-ref-name>

<res-type>javax.sql.DataSource</res-type>

<res-auth>Container</res-auth>

</resource-ref>

</web-app>



