МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Кафедра «Програмна інженерія та інформаційні технології управління»

Звіт з лабораторної роботи №6

З предмету «Об’єктно-орієнтоване програмування»

Виконав

Студент групи КН-36а

Рубан Ю.Д.

Перевірили:

Козуля М.М.

Кізілов О.С.

Харків ­– 2017

ЗАВДАННЯ НА ЛАБОРАТОРНУ РОБОТУ

Модифікувати раніше створену програму індивідуального завдання [лабораторної роботи № 3](http://iwanoff.96.lt/oop_kn/LabTraining03.html), побудувавши код з використанням проектних патернів. Обов'язково реалізувати патерни "Facade", "Singleton" "Lazy Initialization" і "Factory Method". Додатково можна застосувати патерни "Abstract Factory" і "Observer", а також будь-які інші патерни на свій розсуд.

ХІД ВИКОНАННЯ РОБОТИ

Було модифіковано раніше створену програму індивідуального завдання [лабораторної роботи № 3](http://iwanoff.96.lt/oop_kn/LabTraining03.html), побудувавши код з використанням проектних патернів. Реалізовано патерни "Facade", "Singleton" "Lazy Initialization" і " Abstract Factory ".

КОД ПРОГРАМИ

Файл IDay.java

package AbsClasses;  
  
import Model.Day;  
import javafx.beans.property.IntegerProperty;  
import javafx.beans.property.StringProperty;  
  
import javax.xml.bind.annotation.XmlSeeAlso;  
  
@XmlSeeAlso({Day.class})  
public abstract class IDay {  
 abstract public int getCountOfvisitors();  
 abstract public String getComment();  
 abstract public IntegerProperty passengersProperty();  
 abstract public StringProperty commentProperty();  
 abstract public void setComment(String comment);  
 abstract public void setCountOfvisitors(int countOfvisitors);  
}

Файл IDayFactory.java

package AbsClasses;  
  
public abstract class IDayFactory {  
 abstract public IDay createDay(int visitors, String comment);  
}

Файл IShow.java

package AbsClasses;  
  
import Model.Show;  
import javafx.beans.property.StringProperty;  
import javax.xml.bind.annotation.XmlSeeAlso;  
import java.util.List;  
@XmlSeeAlso({Show.class})  
public abstract class IShow {  
 abstract public void add(IDay d);  
 abstract public String dayWithLargeComment();  
 abstract public Integer maxVisitors();  
 abstract public Integer visitors();  
 abstract public void sortViaABC();  
 abstract public void sortViaVisitors();  
 abstract public List<IDay>getDays();  
 abstract public String getTitle();  
 abstract public String getName();  
 abstract public void setName(String name);  
 abstract public void setTitle(String title);  
 abstract public StringProperty getPropertyName();  
 abstract public StringProperty getPropertyTitle();  
}

Файл IShowFactory.java

package AbsClasses;  
  
import Exceptions.InputException;  
  
public abstract class IShowFactory {  
 public abstract IShow createShow(String name, String title) throws InputException;  
}

Файл IShows.java

package AbsClasses;  
  
import Model.Shows;  
  
import javax.xml.bind.annotation.XmlSeeAlso;  
import java.util.List;  
  
@XmlSeeAlso({Shows.class})  
public abstract class IShows {  
 abstract public List<IShow> getList();  
 abstract public void setList(List<IShow> shows);  
  
}

Файл IShowsFactory.java

package AbsClasses;  
  
public abstract class IShowsFactory {  
 abstract public IShows createShows();  
 abstract public IShowFactory getShowFactory();  
 abstract public IDayFactory getDayFactory();  
}

Файл InputException.java

package Exceptions;  
  
public class InputException extends Exception{  
 public String title;  
 public String name;  
 public InputException(String t,String n)  
 {  
 title=t;  
 name=n;  
 }  
}

Файл DayFactory.java

package Factories;  
  
import AbsClasses.IDay;  
import AbsClasses.IDayFactory;  
import Model.Day;  
  
public class DayFactory extends IDayFactory {  
 public IDay createDay(int visitors, String comment) throws IllegalArgumentException  
 {  
 return new Day(visitors,comment);  
 }  
}

Файл ShowFactory.java

package Factories;  
  
import AbsClasses.IShowFactory;  
import Exceptions.InputException;  
import Model.Show;  
import AbsClasses.IShow;  
  
public class ShowFactory extends IShowFactory {  
  
 @Override  
 public IShow createShow(String name, String title) throws InputException {  
 return new Show(name,title);  
 }  
  
}

Файл ShowsFactory.java

package Factories;  
  
import AbsClasses.IDayFactory;  
import AbsClasses.IShowFactory;  
import AbsClasses.IShows;  
import AbsClasses.IShowsFactory;  
import Model.Shows;  
  
public class ShowsFactory extends IShowsFactory {  
 @Override  
 public IShows createShows() {  
 return new Shows();  
 }  
  
 private DayFactory d = new DayFactory();  
 private ShowFactory s = new ShowFactory();  
 @Override  
 public IShowFactory getShowFactory() {  
 return s;  
 }  
  
 @Override  
 public IDayFactory getDayFactory() {  
 return d;  
 }  
}

Файл Day.java

package Model;  
  
  
import AbsClasses.IDay;  
import javafx.beans.property.IntegerProperty;  
import javafx.beans.property.SimpleIntegerProperty;  
import javafx.beans.property.SimpleStringProperty;  
import javafx.beans.property.StringProperty;  
import javax.xml.bind.annotation.XmlElement;  
  
public class Day extends IDay{  
 private IntegerProperty countOfvisitors;  
 private StringProperty comment;  
  
 public IntegerProperty passengersProperty() {  
 return countOfvisitors;  
 }  
 public StringProperty commentProperty() {  
 return comment;  
 }  
 @XmlElement(name = "comment")  
 public String getComment() {  
 return comment.get();  
 }  
 @XmlElement(name = "countOfvisitors")  
 public int getCountOfvisitors() {  
 return countOfvisitors.get();  
 }  
 public void setComment(String comment) {  
 this.comment.set(comment);  
 }  
 public void setCountOfvisitors(Integer countOfvisitors) {  
 if(countOfvisitors<0)throw new IllegalArgumentException();  
 this.countOfvisitors.setValue(countOfvisitors);  
 }  
  
 public Day() {  
 this(1, "null");  
 }  
  
 public Day(int countOfvisitors) {  
 if(countOfvisitors<0)throw new IllegalArgumentException();  
 this.countOfvisitors.set(countOfvisitors);  
 this.comment.set(null);  
 }  
  
 @Override  
 public String toString() {  
 return "\ncountOfvisitors = " + countOfvisitors.get()  
 + " Comment = "+ comment.get();  
 }  
  
 @Override  
 public boolean equals(Object obj) {  
 Day s = (Day) obj;  
 return this.countOfvisitors == s.countOfvisitors && this.comment==s.comment;  
 }  
  
 public Day(int countOfvisitors, String comment) {  
 if(countOfvisitors<0)throw new IllegalArgumentException();  
 this.countOfvisitors = new SimpleIntegerProperty(countOfvisitors);  
 this.comment = new SimpleStringProperty(comment.trim());  
 }  
}

Файл ModelFacade.java

package Model;  
  
import AbsClasses.IShows;  
import AbsClasses.IShow;  
import AbsClasses.IDay;  
import AbsClasses.IShowsFactory;  
import Exceptions.InputException;  
  
import javax.xml.bind.JAXBContext;  
import javax.xml.bind.JAXBException;  
import javax.xml.bind.Marshaller;  
import javax.xml.bind.Unmarshaller;  
import java.io.File;  
import java.util.List;  
  
public class ModelFacade {  
 private IShows shows;  
 private IShowsFactory showsFactory;  
 private static ModelFacade *InstanceFacade* = null;  
  
 private ModelFacade() {  
 }  
  
 public void setShowsFactory(IShowsFactory showsFactory) {  
 this.showsFactory = showsFactory;  
 shows = showsFactory.createShows();  
 }  
  
 public static ModelFacade getInstance() {  
 if (*InstanceFacade* == null) *InstanceFacade* = new ModelFacade();  
 return *InstanceFacade*;  
 }  
  
 public void addShow(IShow show) {  
 shows.getList().add(show);  
 }  
  
 public IShow createShow(String name, String title) throws InputException {  
 return showsFactory.getShowFactory().createShow(name, title);  
 }  
  
 public IDay createDay(int count, String comment)throws IllegalArgumentException  
 {  
 return showsFactory.getDayFactory().createDay(count,comment);  
 }  
 public void dellShow(IShow show)  
 {  
 shows.getList().remove(show);  
 }  
  
 public void addAll(List<IShow> show) {  
 shows.getList().addAll(show);  
  
 }  
  
 public void dellDay(IShow show, IDay day)  
 {  
 shows.getList().get(shows.getList().indexOf(show)).getDays().remove(day);  
 }  
  
 public IShows getShows() {  
 return shows;  
 }  
  
 public void loadDataFromFile(File file) throws JAXBException {  
 if (file != null) {  
 try {  
 JAXBContext context = JAXBContext  
 .*newInstance*(IShows.class);  
 Unmarshaller um = context.createUnmarshaller();  
 this.shows = (IShows) um.unmarshal(file);  
 } catch (Exception e) {  
 throw e;  
  
 }  
 }  
 }  
  
 public void sortABC(IShow show)  
 {  
 shows.getList().get(shows.getList().indexOf(show)).sortViaABC();  
  
 }  
 public void sortVisitors(IShow show)  
 {  
 shows.getList().get(shows.getList().indexOf(show)).sortViaVisitors();  
  
 }  
 public void saveToXML(File file) throws JAXBException {  
  
 if (file != null) {  
 try {  
 JAXBContext context = JAXBContext  
 .*newInstance*(IShows.class);  
 Marshaller marshaller = context.createMarshaller();  
 marshaller.setProperty(Marshaller.*JAXB\_FORMATTED\_OUTPUT*, true);  
 marshaller.marshal(shows,file);  
 } catch (Exception e) {  
 throw e;  
 }  
 }  
 }  
}

Файл Show.java

package Model;  
import AbsClasses.IShow;  
import AbsClasses.IDay;  
import javafx.beans.property.SimpleStringProperty;  
import javafx.beans.property.StringProperty;  
import Exceptions.InputException;  
  
import javax.xml.bind.annotation.XmlElement;  
import java.util.ArrayList;  
import java.util.Collections;  
  
  
public class Show extends IShow{  
 private StringProperty title;  
 private StringProperty name;  
  
 private ArrayList<IDay> days = new ArrayList<>();  
 public void sortViaVisitors()  
 {  
 Collections.*sort*(days,(IDay o1, IDay o2)->(Integer.*compare*(o2.getCountOfvisitors(),o1.getCountOfvisitors())));  
 }  
 public void sortViaABC()  
 {  
 Collections.*sort*(days,(IDay o1,IDay o2)->o1.getComment().compareTo(o2.getComment()));  
 }  
 @Override  
 public String toString() {  
 return "Show{" +  
 "title='" + title.get() + '\'' +  
 ", name='" + name.get() + '\'' + days.toString() + '}';  
 }  
 @XmlElement(name = "name")  
 public String getName() {  
 return name.get();  
 }  
 @XmlElement(name = "title")  
 public String getTitle() {  
 return title.get();  
 }  
 @XmlElement(name = "days")  
 public ArrayList<IDay> getDays() {  
 return days;  
 }  
  
 public StringProperty getPropertyName() {  
 return name;  
 }  
 public StringProperty getPropertyTitle() {  
 return title;  
 }  
 public void setTitle(String title) {  
 this.title.set(title.trim());  
 }  
  
 public void setName(String name) {  
 this.name.set(name.trim());  
 }  
  
 public void setDays(ArrayList<IDay> days) {  
 this.days = days;  
 }  
  
 public void add(IDay ws) {  
 boolean exist = false;  
 for (IDay s:  
 days) {  
 if(s.equals(ws)) {  
 exist = true;  
 break;  
 }  
 }  
 if(!exist) days.add(ws);  
 }  
 public Integer visitors()  
 {  
 Integer count = 0;  
 for(IDay d : days)  
 {  
 count+=d.getCountOfvisitors();  
 }  
 return count;  
 }  
 public Integer maxVisitors()  
 {  
 int max = 0;  
 for(IDay d : days)  
 {  
 if(max<d.getCountOfvisitors())  
 {  
 max = d.getCountOfvisitors();  
 }  
 }  
 return max;  
 }  
 public String dayWithLargeComment()  
 {  
 int max = 0;  
 int index=0;  
 for(int i =0;i<days.size();i++)  
 {  
 if(max<days.get(i).getComment().length())  
 {  
 max = days.get(i).getComment().length();  
 index = i;  
 }  
 }  
 return days.get(index).getComment();  
 }  
 public boolean search(IDay ws) {  
 for (IDay day: days) {  
 if(day.equals(ws)){  
 return true;  
 }  
 }  
 return false;  
 }  
  
 public Integer totalDays() {  
 return days.size();  
 }  
 public IDay getCountOfvisitors(int i) {  
 return days.get(i);  
 }  
 public Show() throws InputException {  
 this("null", "null");  
 }  
  
 public Show(String title, String name)throws InputException {  
 if(title.trim().length()==0 || name.trim().length()==0)throw new InputException(title,name);  
 this.title = new SimpleStringProperty(title.trim());  
 this.name = new SimpleStringProperty(name.trim());  
 }  
}

Файл Shows.java

package Model;  
  
import AbsClasses.IShows;  
import AbsClasses.IShow;  
  
import javax.xml.bind.annotation.XmlElement;  
import javax.xml.bind.annotation.XmlRootElement;  
import java.util.ArrayList;  
import java.util.List;  
  
@XmlRootElement(name="Shows")  
public class Shows extends IShows {  
 List<IShow>shows = new ArrayList<>();  
 public Shows() {shows = new ArrayList<>(); }  
  
 @XmlElement(name = "Show")  
 public List<IShow> getList() {  
 return shows;  
 }  
  
 @Override  
 public String toString() {  
 String res = "";  
 for (IShow ts:  
 shows) {  
 res+= "\nTitle: " + ts.getTitle()  
 + "\nName: " + ts.getName() + "\n"+ ts.getDays();  
 }  
 return res;  
 }  
  
 public void setList(List<IShow> ws) {  
 shows = ws;  
 }  
}

Файл Controller.java

package sample;  
  
import AbsClasses.\*;  
import Exceptions.InputException;  
import Factories.ShowsFactory;  
import Model.ModelFacade;  
import javafx.collections.FXCollections;  
import javafx.collections.ObservableList;  
import javafx.fxml.FXML;  
import javafx.scene.control.Alert;  
import javafx.scene.control.TableColumn;  
import javafx.scene.control.TableView;  
import javafx.scene.control.TextField;  
import javafx.scene.control.cell.TextFieldTableCell;  
import javafx.stage.FileChooser;  
import javafx.util.converter.NumberStringConverter;  
  
import javax.xml.bind.JAXBException;  
import java.io.File;  
  
  
public class Controller {  
 private ModelFacade facade;  
 Main main;  
 @FXML  
 private TextField addTitle;  
 @FXML  
 private TextField addName;  
 @FXML  
 private TextField addVisitors;  
 @FXML  
 private TextField addComment;  
 @FXML  
 private TableView<IShow> ShowTableView;  
 @FXML  
 private TableView<IDay> DayTableView;  
 @FXML  
 private TableColumn<IShow, String> titleSH;  
 @FXML  
 private TableColumn<IShow, String> nameSH;  
 @FXML  
 private TableColumn<IDay, Number> visitorsNum;  
 @FXML  
 private TableColumn<IDay, String> commentN;  
  
 public void initialize() {  
 titleSH.setCellFactory(TextFieldTableCell.*forTableColumn*());  
 titleSH.setCellValueFactory(cellBack -> cellBack.getValue().getPropertyTitle());  
 titleSH.setOnEditCommit(event -> event.getTableView().getItems().get(event.getTablePosition().getRow()).setTitle(event.getNewValue()));  
 nameSH.setCellValueFactory(cellBack -> cellBack.getValue().getPropertyName());  
 nameSH.setCellFactory(TextFieldTableCell.*forTableColumn*());  
 nameSH.setOnEditCommit(  
 event -> (event.getTableView().getItems().get(  
 event.getTablePosition().getRow())  
 ).setName(event.getNewValue())  
 );  
  
 showDetails(null);  
  
 visitorsNum.setCellFactory(TextFieldTableCell.*forTableColumn*(new NumberStringConverter()));  
 visitorsNum.setOnEditCommit(  
 event -> (event.getTableView().getItems().get(  
 event.getTablePosition().getRow())  
 ).setCountOfvisitors(event.getNewValue().intValue())  
 );  
 commentN.setCellFactory(TextFieldTableCell.*forTableColumn*());  
 commentN.setOnEditCommit(  
 event -> (event.getTableView().getItems().get(  
 event.getTablePosition().getRow())  
 ).setComment(event.getNewValue())  
 );  
  
 ShowTableView.getSelectionModel().selectedItemProperty().addListener(  
 (observable, old, selected) -> showDetails(selected));  
 facade = ModelFacade.*getInstance*();  
 facade.setShowsFactory(new ShowsFactory());  
 }  
  
  
 private void showDetails(IShow show) {  
 if (show != null) {  
 visitorsNum.setCellValueFactory(cellBack -> cellBack.getValue().passengersProperty());  
 commentN.setCellValueFactory(cellBack -> cellBack.getValue().commentProperty());  
 ObservableList<IDay> temp = FXCollections.*observableArrayList*();  
 temp.addAll(show.getDays());  
 DayTableView.setItems(temp);  
 }  
 }  
  
 public void setMain(Main main) {  
 this.main = main;  
 ObservableList<IShow> temp = FXCollections.*observableArrayList*();  
 temp.addAll( facade.getShows().getList());  
 ShowTableView.setItems(temp);  
 }  
  
 @FXML  
 private void handleSaveAs() {  
 FileChooser fileChooser = new FileChooser();  
 fileChooser.setTitle("Save Document");  
 FileChooser.ExtensionFilter extFilter =  
 new FileChooser.ExtensionFilter("XML files (\*.xml)", "\*.xml");  
 fileChooser.getExtensionFilters().add(extFilter);  
 File file = fileChooser.showSaveDialog(main.getStage());  
 try {  
 facade.saveToXML(file);  
 } catch (JAXBException e) {  
 Alert alert = new Alert(Alert.AlertType.*ERROR*);  
 alert.setTitle("Error");  
 alert.setHeaderText("Could not save data");  
 alert.setContentText("Could not save data to file:\n" + file.getPath());  
  
 alert.showAndWait();  
 }  
 }  
 @FXML  
 private void handleAddShow() {  
try {  
 facade.addShow(facade.createShow(addTitle.getText(), addName.getText()));  
 ObservableList<IShow> temp = FXCollections.*observableArrayList*();  
 temp.addAll((facade.getShows().getList()));  
 ShowTableView.setItems(temp);  
 addTitle.clear();  
 addName.clear();  
}catch (InputException e)  
{  
 Alert alert = new Alert(Alert.AlertType.*ERROR*);  
 alert.setTitle("Error");  
 alert.setHeaderText("Cant create show without name or/and name");  
 if (e.name=="" && e.title!="")  
 alert.setContentText("name is null");  
 else if(e.title=="" && e.name!="")  
 alert.setContentText("name is null");  
 else alert.setContentText("name and title are null");  
 alert.showAndWait();  
  
}  
 }  
 @FXML  
 private void handleAbout() {  
 Alert alert = new Alert(Alert.AlertType.*INFORMATION*);  
 alert.setTitle("NewApp");  
 alert.setHeaderText("About: version 1.6.0");  
 alert.setContentText("Author: Yura Ruban\n");  
 alert.showAndWait();  
 }  
 @FXML  
 private void handleAddDay() {  
 try {  
 IShow selected = ShowTableView.getSelectionModel().selectedItemProperty().get();  
 selected.add(facade.createDay(Integer.*parseInt*(addVisitors.getText()),  
 String.*valueOf*(addComment.getText())));  
 ObservableList<IDay> temp = FXCollections.*observableArrayList*();  
 temp.addAll( selected.getDays());  
 DayTableView.setItems(FXCollections.*observableArrayList*(temp));  
 addVisitors.clear();  
 addComment.clear();  
 showDetails(selected);  
 }catch (IllegalArgumentException e)  
 {  
 Alert alert = new Alert(Alert.AlertType.*ERROR*);  
 alert.setTitle("Error");  
 alert.setHeaderText("Wrong number");  
 alert.setContentText("Those number is not valid");  
  
 alert.showAndWait();  
  
 }  
 }  
 @FXML  
 private void handleOpen() {  
 FileChooser fileChooser = new FileChooser();  
 FileChooser.ExtensionFilter extFilter = new FileChooser.ExtensionFilter(  
 "XML files (\*.xml)", "\*.xml");  
 fileChooser.getExtensionFilters().add(extFilter);  
  
 File file = fileChooser.showOpenDialog(main.getStage());  
  
 if (file != null) {  
 try {  
 facade.loadDataFromFile(file);  
 } catch (JAXBException e) {  
 Alert alert = new Alert(Alert.AlertType.*ERROR*);  
 alert.setTitle("Error");  
 alert.setHeaderText("Could not load data");  
 alert.setContentText("Could not load data from file:\n" + file.getPath());  
  
 alert.showAndWait();  
 }  
 ObservableList<IShow> temp = FXCollections.*observableArrayList*();  
 temp.clear();  
 temp.addAll(facade.getShows().getList());  
 ShowTableView.setItems(temp);  
 }  
  
 }  
  
 @FXML  
 public void handleDelete(){  
 IShow show = ShowTableView.getSelectionModel().selectedItemProperty().get();  
 IDay day = DayTableView.getSelectionModel().selectedItemProperty().get();  
 if(day == null){  
 facade.dellShow(show);  
 ObservableList<IShow> temp = FXCollections.*observableArrayList*();  
 temp.addAll(facade.getShows().getList());  
 DayTableView.setItems(null);  
 ShowTableView.setItems(temp);  
 }else{  
 facade.dellDay(show,day);  
 ObservableList<IDay> temp = FXCollections.*observableArrayList*();  
 temp.addAll(show.getDays());  
 DayTableView.setItems(temp);  
 }  
 }  
  
 @FXML  
 public void handleVisitorSort()  
 {  
 int index = ShowTableView.getSelectionModel().getSelectedIndex();  
 IShow temp = ShowTableView.getItems().get(index);  
 facade.sortVisitors(temp);  
 DayTableView.setItems(FXCollections.*observableArrayList*(temp.getDays()));  
 }  
 @FXML  
 public void handleABCSort()  
 {  
 int index = ShowTableView.getSelectionModel().getSelectedIndex();  
 IShow temp = ShowTableView.getItems().get(index);  
 facade.sortABC(temp);  
 DayTableView.setItems(FXCollections.*observableArrayList*((temp.getDays())));  
 }  
 @FXML  
 public void handleShowSumCount()  
 {  
 int index = ShowTableView.getSelectionModel().getSelectedIndex();  
 IShow temp = ShowTableView.getItems().get(index);  
 Alert alert = new Alert(Alert.AlertType.*INFORMATION*);  
 alert.setHeaderText("visitors");  
 alert.setTitle("visitors");  
 alert.setContentText(temp.visitors().toString());  
 alert.show();  
 }  
 @FXML  
 public void handleMaxVisitors()  
 {  
 int index = ShowTableView.getSelectionModel().getSelectedIndex();  
 IShow temp = ShowTableView.getItems().get(index);  
 Alert alert = new Alert(Alert.AlertType.*INFORMATION*);  
 alert.setHeaderText("Max Visitors");  
 alert.setTitle("Max Visitors");  
 alert.setContentText(temp.maxVisitors().toString());  
 alert.show();  
 }  
 @FXML  
 public void handleLargeComment()  
 {  
 int index = ShowTableView.getSelectionModel().getSelectedIndex();  
 IShow temp = ShowTableView.getItems().get(index);  
 Alert alert = new Alert(Alert.AlertType.*INFORMATION*);  
 alert.setHeaderText("Large Comment");  
 alert.setTitle("Large Comment");  
 alert.setContentText(temp.dayWithLargeComment());  
 alert.show();  
 }  
}

Файл Main.java

package sample;  
import javafx.application.Application;  
import javafx.fxml.FXMLLoader;  
import javafx.geometry.Pos;  
import javafx.scene.Scene;  
import javafx.scene.control.Alert;  
import javafx.scene.control.Button;  
import javafx.scene.control.Label;  
import javafx.scene.layout.AnchorPane;  
import javafx.scene.layout.VBox;  
import javafx.stage.FileChooser;  
import javafx.stage.Modality;  
import javafx.stage.Stage;  
import javax.xml.bind.JAXBContext;  
import javax.xml.bind.JAXBException;  
import javax.xml.bind.Marshaller;  
import javax.xml.bind.Unmarshaller;  
import java.io.File;  
import java.io.IOException;  
import java.util.ArrayList;  
import java.util.List;  
  
public class Main extends Application {  
 private Stage primaryStage;  
 private AnchorPane root;  
  
 public Stage getStage() {  
 return primaryStage;  
 }  
  
 @Override  
 public void start(Stage primaryStage) throws Exception{  
 this.primaryStage = primaryStage;  
 primaryStage.setOnCloseRequest(e -> {  
 if(!display("Exit", "Do you want to exit?")) e.consume();  
 });  
 this.primaryStage.setTitle("Shows");  
 initRootLayout();  
  
 }  
  
 public void initRootLayout() {  
 try {  
 FXMLLoader loader = new FXMLLoader();  
 loader.setLocation(Main.class  
 .getResource("sample.fxml"));  
 root = (AnchorPane) loader.load();  
  
 Scene scene = new Scene(root);  
 primaryStage.setScene(scene);  
  
 Controller controller = loader.getController();  
 controller.setMain(this);  
  
 primaryStage.show();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
  
  
  
  
  
 boolean answer;  
  
 public boolean display(String title, String message) {  
 Stage window = new Stage();  
 window.initModality(Modality.*APPLICATION\_MODAL*);  
 window.setTitle(title);  
 window.setMinWidth(250);  
 Label label = new Label();  
 label.setText(message);  
  
 Button yesButton = new Button("Yes");  
 Button noButton = new Button("No");  
  
 yesButton.setOnAction(e -> {  
 answer = true;  
 window.close();  
 });  
 noButton.setOnAction(e -> {  
 answer = false;  
 window.close();  
 });  
  
 VBox layout = new VBox(10);  
 layout.getChildren().addAll(label, yesButton, noButton);  
 layout.setAlignment(Pos.*CENTER*);  
 Scene scene = new Scene(layout);  
 window.setScene(scene);  
 window.showAndWait();  
 return answer;  
 }  
  
 public static void main(String[] args) {  
 *launch*(args);  
 }  
}

Файл sample.fxml

<?xml version="1.0" encoding="UTF-8"?>  
  
<?import javafx.scene.control.Button?>  
<?import javafx.scene.control.ButtonBar?>  
<?import javafx.scene.control.Label?>  
<?import javafx.scene.control.Menu?>  
<?import javafx.scene.control.MenuBar?>  
<?import javafx.scene.control.MenuItem?>  
<?import javafx.scene.control.SplitPane?>  
<?import javafx.scene.control.TableColumn?>  
<?import javafx.scene.control.TableView?>  
<?import javafx.scene.control.TextField?>  
<?import javafx.scene.layout.AnchorPane?>  
  
<AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="400.0" prefWidth="600.0" xmlns="http://javafx.com/javafx/8.0.112" xmlns:fx="http://javafx.com/fxml/1" fx:controller="sample.Controller">  
 <children>  
 <SplitPane dividerPositions="0.4816053511705686" layoutX="133.0" layoutY="24.0" prefHeight="376.0" prefWidth="600.0" AnchorPane.bottomAnchor="0.0" AnchorPane.leftAnchor="0.0" AnchorPane.rightAnchor="0.0" AnchorPane.topAnchor="24.0">  
 <items>  
 <AnchorPane minHeight="0.0" minWidth="0.0" prefHeight="398.0" prefWidth="249.0">  
 <children>  
 <TableView fx:id="ShowTableView" editable="true" layoutX="8.0" layoutY="40.0" prefHeight="308.0" prefWidth="285.0" AnchorPane.bottomAnchor="66.0" AnchorPane.leftAnchor="0.0" AnchorPane.rightAnchor="0.0" AnchorPane.topAnchor="0.0">  
 <columns>  
 <TableColumn fx:id="titleSH" prefWidth="140.0" text="Title" />  
 <TableColumn fx:id="nameSH" minWidth="0.0" prefWidth="144.0" text="Name" />  
 </columns>  
 </TableView>  
 <TextField fx:id="addTitle" layoutX="64.0" layoutY="308.0" prefHeight="25.0" prefWidth="137.0" />  
 <Label layoutX="14.0" layoutY="312.0" text="Title" />  
 <Label layoutX="14.0" layoutY="340.0" text="Name" />  
 <TextField fx:id="addName" layoutX="64.0" layoutY="336.0" prefHeight="25.0" prefWidth="137.0" />  
 <Button layoutX="221.0" layoutY="308.0" mnemonicParsing="false" onAction="#handleAddShow" prefHeight="49.0" prefWidth="38.0" text="Add" />  
 </children>  
 </AnchorPane>  
 <AnchorPane minHeight="0.0" minWidth="0.0" prefHeight="160.0" prefWidth="100.0">  
 <children>  
 <ButtonBar buttonMinWidth="55.0" layoutX="145.0" layoutY="334.0" AnchorPane.leftAnchor="110.0" AnchorPane.rightAnchor="60.0" AnchorPane.topAnchor="334.0" />  
 <TableView fx:id="DayTableView" editable="true" layoutX="-1.0" layoutY="31.0" prefHeight="308.0" prefWidth="307.0" AnchorPane.bottomAnchor="66.0" AnchorPane.leftAnchor="-1.0" AnchorPane.rightAnchor="1.0" AnchorPane.topAnchor="0.0">  
 <columns>  
 <TableColumn fx:id="visitorsNum" prefWidth="92.0" text="Visitors" />  
 <TableColumn fx:id="commentN" prefWidth="214.0" text="Comment" />  
 </columns>  
 </TableView>  
 <Label layoutX="14.0" layoutY="317.0" text="Visitors" />  
 <TextField fx:id="addVisitors" layoutX="72.0" layoutY="313.0" prefHeight="25.0" prefWidth="80.0" />  
 <Button layoutX="160.0" layoutY="313.0" mnemonicParsing="false" onAction="#handleAddDay" text="Add" />  
 <Button layoutX="209.0" layoutY="313.0" mnemonicParsing="false" onAction="#handleDelete" text="Delete" />  
 <Label layoutX="13.0" layoutY="348.0" prefHeight="17.0" prefWidth="80.0" text="Comment" />  
 <TextField fx:id="addComment" layoutX="72.0" layoutY="344.0" prefHeight="25.0" prefWidth="188.0" />  
 </children>  
 </AnchorPane>  
 </items>  
 </SplitPane>  
 <MenuBar layoutY="2.0" AnchorPane.bottomAnchor="375.0" AnchorPane.leftAnchor="0.0" AnchorPane.rightAnchor="0.0" AnchorPane.topAnchor="0.0">  
 <menus>  
 <Menu mnemonicParsing="false" text="File">  
 <items>  
 <MenuItem mnemonicParsing="false" onAction="#handleOpen" text="Open" />  
 <MenuItem mnemonicParsing="false" onAction="#handleSaveAs" text="Save" />  
 </items>  
 </Menu>  
 <Menu mnemonicParsing="false" text="Help">  
 <items>  
 <MenuItem mnemonicParsing="false" onAction="#handleAbout" text="About" />  
 </items>  
 </Menu>  
 <Menu mnemonicParsing="false" text="Sort &amp; Search">  
 <items>  
 <MenuItem mnemonicParsing="false" onAction="#handleVisitorSort" text="SortViaVisitors" />  
 <MenuItem mnemonicParsing="false" onAction="#handleABCSort" text="SortViaABC" />  
 <MenuItem mnemonicParsing="false" onAction="#handleShowSumCount" text="Show count of visitors" />  
 <MenuItem mnemonicParsing="false" onAction="#handleMaxVisitors" text="Show max visitors" />  
 <MenuItem mnemonicParsing="false" onAction="#handleLargeComment" text="largest comment" />  
 </items>  
 </Menu>  
 </menus>  
 </MenuBar>  
 </children>  
</AnchorPane>

Результат виконання програми показаний на рисунку 1.1

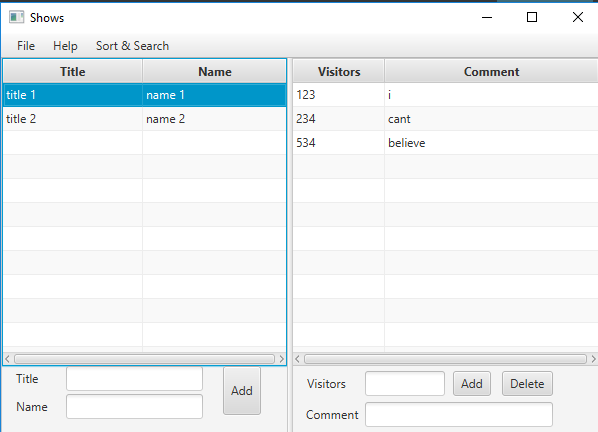


Рисунок 1.1 – Інтерфейс програми

ВИСНОВКИ

В даній лабораторній роботі було вивчено теоретичний матеріал з теми «Паттерни проектування». Були отримані практичні навички з використання паттернів Façade, Singleton, Abstract Factory, Lazy initialization та Factory Method.