Примеры программной реализации модели, вида и контроллера

Рассмотрим пример решения задачи в рамках MVC. Пусть необходимо хранить информацию о некоторой организации, работникам которой выделяется премиальный фонд по случаю календарного праздника. Определим простую функциональность приложения, как возможность пользователю увеличивать премиальный фонд пошагово на одну денежную единицу. В следующих практиках будем изменять и дополнять функциональность и добавлять новые компоненты и технологии в приложение.

Модель содержит поля для хранения информации об организации:

* название (String name);
* количество работников (int personnel);
* название праздника (String holiday);
* дата выплаты премии (LocalDate date);
* премиальный фонд (double bonus).

Приведём полный код класса модели:

import java.time.LocalDate;

public class Organization {

private String name;

private int personnel;

private String holiday;

private LocalDate date;

private double bonus;

public Organization(String name, int personnel, String holiday, LocalDate date, double bonus){

this.name = name;

this.personnel =personnel;

this.holiday = holiday;

this.date = date;

this.bonus = bonus;}

public String getName() {

return name; }

public int getPersonnel(){

return personnel; }

public String getHoliday() {

return holiday; }

public LocalDate getDate () {

return date; }

public double getBonus() {

return bonus; }

public void increaseBonus() {

bonus++; }}

Задача класса вида создать элемент-контейнер для размещения на сцене, способный отображать информацию о модели. Важная часть этого класса – элементы для организации взаимодействия с моделью: поле-ссылка на объект-модель (Organization org), метод для изменения состояния вида, если в модели произошли изменения (setInform()) и метод, позволяющий заменить объект модели, отображаемый в этом виде (setOrganization (Organization org)).

Код класса вид:

import java.time.format.DateTimeFormatter;

import javafx.geometry.HPos;

import javafx.geometry.Insets;

import javafx.geometry.Pos;

import javafx.scene.control.Label;

import javafx.scene.layout.GridPane;

import javafx.scene.text.Font;

import javafx.scene.text.FontWeight;

import javafx.scene.text.Text;

public class ViewOrganization {

private Organization org;

private GridPane grid;

private Text nameOrg;

private Text holidayOrg;

private Text cashBonus;

private void createPane(){

grid = new GridPane();

grid.setAlignment(Pos.*CENTER*);

grid.setHgap(10);

grid.setVgap(10);

grid.setPadding(new Insets(25, 25, 25, 25));

nameOrg = new Text();

nameOrg.setFont(Font.*font*("Tahoma", FontWeight.*NORMAL*, 20));

GridPane.*setHalignment*(nameOrg, HPos.*CENTER*);

grid.add(nameOrg, 0, 0, 2, 1);

holidayOrg = new Text();

holidayOrg.setFont(Font.*font*("Tahoma", FontWeight.*NORMAL*, 20));

grid.add(holidayOrg, 0, 1, 2, 1);

Label cashBonusTitle = new Label("Bonus");

cashBonusTitle.setFont(Font.*font*("Tahoma", FontWeight.*NORMAL*, 20));

grid.add(cashBonusTitle, 0, 2);

cashBonus = new Text();

cashBonus.setFont(Font.*font*("Tahoma", FontWeight.*NORMAL*, 20));

grid.add(cashBonus, 1, 2);}

public GridPane getPane() {

return grid; }

public void setOrganization (Organization org) {

this.org = org;

setInform(); }

public void setInform() {

nameOrg.setText(org.getName());

holidayOrg.setText(org.getHoliday()+" - " +org.getDate().format(DateTimeFormatter.*ofPattern*("dd.MM.uuuu")));

cashBonus.setText(Double.*toString*(org.getBonus())); }

public ViewOrganization(Organization org) {

createPane();

setOrganization(org); }}

Контроллером будет являться главный класс приложения. Рассмотрим его подробно.

Контроллер создаёт объект-модель (Organization org), объект-вид (ViewOrganization viewOrg) и элементы для взаимодействия с пользователем (Button btn). В данном случае это кнопка, с помощью которой пользователь увеличивает премиальный фонд. Контроллер инициирует изменение модели и вида в коде обработчика действия кнопки. Приведём полный код главного класса приложения:

import javafx.application.Application;

import javafx.geometry.Pos;

import javafx.scene.Scene;

import javafx.scene.control.Button;

import javafx.scene.layout.VBox;

import javafx.stage.Stage;

import java.time.LocalDate;

public class Main extends Application {

@Override

public void start(Stage primaryStage) {

primaryStage.setTitle("Simple Model-View");

Organization org = new Organization("Horns&Hoof",10,"International Women's Day", LocalDate.*of*(2020, 3, 6),0);

VBox root = new VBox(10);

root.setAlignment(Pos.*CENTER*);

ViewOrganization viewOrg = new ViewOrganization(org);

root.getChildren().add(viewOrg.getPane());

Button btn = new Button("+");

btn.setPrefSize(50, 50);

btn.setOnAction((event) -> {

org.increaseBonus();

viewOrg.setInform(); });

root.getChildren().add(btn);

Scene scene = new Scene(root, 500, 300);

primaryStage.setScene(scene);

primaryStage.show(); }

public static void main(String[] args) {

*launch*(args); }}