

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
УЧРЕЖДЕНИЯ ОБРАЗОВАНИЯ
«БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

ФАКУЛЬТЕТ ЭЛЕКТРОННО-ИНФОРМАЦИОННЫХ СИСТЕМ

Кафедра интеллектуальных информационных технологий

Отчет по лабораторной работе №10

Специальность ПОЗ

Выполнила
Гаврилюк Р. И., студентка
группы ПОЗ

Проверил
Крощенко А. А., ст. преп.
Кафедры ИИТ

Брест 2021

Вариант 6

Цель работы: приобрести практические навыки разработки многооконных приложений на JavaFX для работы с базами данных.

Задание:

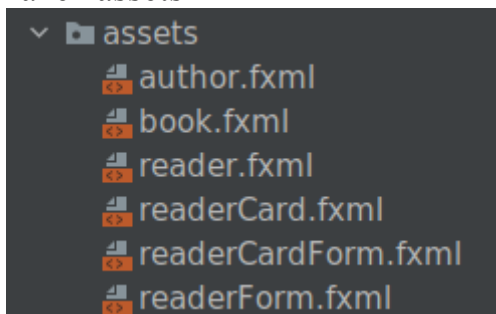
На основе БД, разработанной в лабораторной работе No9, реализовать многооконное приложение клиент, позволяющее выполнять основные операции над таблицей в БД (добавление, удаление, модификацию данных).

б) База данных «Библиотека»

Текст программы:

Пакет main отвечает за отображение бд в формах, обработку событий форм.

Пакет assets



форма для создания автора

форма для создания/редактирования книги

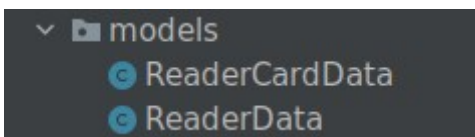
форма для отображения сведений о читателе

форма для отображения карточки читателя

для создания/ред-ия записи в карточке читателя

форма для создания/редактирования читателя

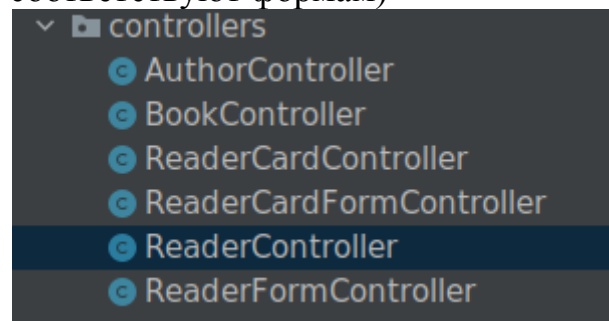
Пакет models



класс для отображения всех данных для карточки читателя

класс для отображения информации о читателях

Пакет controllers содержит контроллеры для обработки событий форм (названия соответствуют формам)



Пример кода контроллеров:

ReaderController.java

```
public class ReaderController {  
  
    @FXML  
    private TableView<ReaderData> readerTable;  
    @FXML  
    private TableColumn<ReaderData, String> FIOColumn;
```

```

@FXML
private TableColumn<ReaderData, String> readerCardColumn;

@FXML
private Label nameLabel;
@FXML
private Label surnameLabel;
@FXML
private Label middleNameLabel;
@FXML
private Label readerCardIdLabel;

@FXML
private TextField findTextField;

private ReaderData currentReaderData;

public ReaderController() { }

@FXML
private void initialize() throws Exception {
    // Инициализация таблицы адресатов с двумя столбцами.
    FIOColumn.setCellValueFactory(cellData -> cellData.getValue().FIOProperty() );
    readerCardColumn.setCellValueFactory(cellData ->
cellData.getValue().readerCardIdProperty());

    refresh();

    showPersonDetails(null);

    readerTable.getSelectionModel().selectedItemProperty().addListener(
        (observable, oldValue, newValue) -> showPersonDetails(newValue));
}

public ReaderData getCurrentReaderData() {
    return currentReaderData;
}

public void setCurrentReaderData(ReaderData currentReaderData) {
    this.currentReaderData = currentReaderData;
}

public void initializeTable(boolean search) throws Exception{
    ObservableList<ReaderData> personData = FXCollections.observableArrayList();

    PgsqlFactory.createDataConnection();
    Connection conn = WrapperConnection.createConnection();

    // 1. init DAO
    ReaderDAO readerDAO = new ReaderDAO(conn);
    ReaderHistoryDAO readerHistoryDAO = new ReaderHistoryDAO(conn);

    // 2. transaction initialization for DAO objects
    EntityTransaction transaction = new EntityTransaction();
    transaction.initTransaction(readerDAO, readerHistoryDAO);
}

```

```

// 3. query execution
try {
    List<Reader> readers = readerDAO.findAll();
    transaction.commit();
    for (Reader reader : readers) {
        if(search) {
            if((reader.getSurname() + reader.getName() +
reader.getMiddleName()).contains(findTextField.getText())) {
                personData.add(new ReaderData(reader.getId(), reader.getName(),
reader.getSurname(),
                    reader.getMiddleName(), reader.getReaderCard()));
            }
        }
        else {
            personData.add(new ReaderData(reader.getId(), reader.getName(),
reader.getSurname(),
                reader.getMiddleName(), reader.getReaderCard()));
        }
    }
} catch (DAOExtension e) {
    transaction.rollback();
    throw new Exception(e);
} finally {
    // 4. transaction closing
    transaction.endTransaction();
}

conn.close();

readerTable.setItems(personData);
}

private void refresh() throws Exception {
    initializeTable(false);
}

private void showPersonDetails(ReaderData readerData) {
    setCurrentReaderData(readerData);
    if (readerData != null) {
        nameLabel.setText(readerData.getName());
        surnameLabel.setText(readerData.getSurname());
        middleNameLabel.setText(readerData.getMiddleName());
        readerCardIdLabel.setText(readerData.getReaderCardId());
    } else {
        nameLabel.setText("");
        surnameLabel.setText("");
        middleNameLabel.setText("");
        readerCardIdLabel.setText("");
    }
}

public void deleteSelectedReader() throws Exception{
    PgsqlFactory.createDataConnection();
    Connection conn = WrapperConnection.createConnection();

    // 1. init DAO
    ReaderDAO readerDAO = new ReaderDAO(conn);

    // 2. transaction initialization for DAO objects

```

```

EntityTransaction transaction = new EntityTransaction();
transaction.initTransaction(readerDAO);

// 3. query execution
try {
    readerDAO.deleteEntityById(currentReaderData.getReaderId());
    transaction.commit();
} catch (DAOExtension e) {
    transaction.rollback();
    throw new Exception(e);
} finally {
    // 4. transaction closing
    transaction.endTransaction();
}

conn.close();

refresh();
}

public void findReaders() throws Exception{
    initializeTable(true);
}

public void openReaderCardForm() {
    try {
        ReaderCardController.currentReaderData = new
ReaderData(currentReaderData.getReaderId(), currentReaderData.getName(),
            currentReaderData.getSurname(), currentReaderData.getMiddleName(),
currentReaderData.getReaderCardId());

        FXMLLoader fxmlLoader = new
FXMLLoader(getClass().getResource("../assets/readerCard.fxml"));
        Parent root = (Parent) fxmlLoader.load();
        Stage stage = new Stage();
        stage.setScene(new Scene(root));
        stage.showAndWait();
        refresh();
    } catch (Exception e) {
        e.printStackTrace();
    }
}

public void createReader() {
    ReaderFormController.setSTATE(false);
    ReaderFormController.setReader(null);
    openReaderForm();
}

public void updateReader() {
    ReaderFormController.setSTATE(true);
    ReaderFormController.setReader(new Reader(currentReaderData.getReaderId(),
currentReaderData.getName(),
        currentReaderData.getSurname(), currentReaderData.getMiddleName(),
currentReaderData.getReaderCardId()));
    openReaderForm();
}

private void openReaderForm() {

```

```

    try {
        FXMLLoader fxmLoader = new
FXMLLoader(getClass().getResource("../assets/readerForm.fxml"));
        Parent root = (Parent) fxmLoader.load();
        Stage stage = new Stage();
        stage.setScene(new Scene(root));
        stage.showAndWait();
        refresh();
    } catch (Exception e) {
        e.printStackTrace();
    }
}
}

```

ReaderFormController.java

```

public class ReaderFormController {
    @FXML
    private TextField nameTextField;
    @FXML
    private TextField surnameTextField;
    @FXML
    private TextField middleNameTextField;
    @FXML
    private TextField numberCardTextField;
    @FXML
    private Button createUpdateButton;
    @FXML
    private Label titleLabel;

    private static Reader reader;
    private static boolean STATE; //true - update, false - insert

    public static void setReader(Reader reader) {
        ReaderFormController.reader = reader;
    }

    public static void setSTATE(boolean STATE) {
        ReaderFormController.STATE = STATE;
    }

    @FXML
    public void initialize() {
        if (STATE) {
            nameTextField.setText(reader.getName());
            surnameTextField.setText(reader.getSurname());
            middleNameTextField.setText(reader.getMiddleName());
            numberCardTextField.setText(reader.getReaderCard());
            createUpdateButton.setText("Редактировать");
            titleLabel.setText("Редактировать читателя");
        }
    }

    public void createOrUpdateReader() throws Exception {
        PgsqlFactory.createDataConnection();
        Connection conn = WrapperConnection.createConnection();

        // 1. init DAO
        ReaderDAO readerDAO = new ReaderDAO(conn);
    }
}

```

```

// 2. transaction initialization for DAO objects
EntityTransaction transaction = new EntityTransaction();
transaction.initTransaction(readerDAO);

// 3. query execution
try {
    if (STATE) {
        readerDAO.updateEntity(new Reader(reader.getId(), nameTextField.getText(),
surnameTextField.getText(),
        middleNameTextField.getText(), numberCardTextField.getText()));

        ReaderCardController.currentReaderData = new ReaderData(reader.getId(),
nameTextField.getText(), surnameTextField.getText(),
        middleNameTextField.getText(), numberCardTextField.getText());
    }
    else {
        readerDAO.createEntity(new Reader(nameTextField.getText(),
surnameTextField.getText(),
        middleNameTextField.getText(), numberCardTextField.getText()));
    }
} catch (DAOExtension e) {
    transaction.rollback();
    throw new Exception(e);
} finally {
    // 4. transaction closing
    transaction.endTransaction();
}

conn.close();

closeWindow();
}

public void cancelChanges() {
    closeWindow();
}

public void closeWindow() {
    Stage stage = (Stage) numberCardTextField.getScene().getWindow();
    stage.close();
}
}

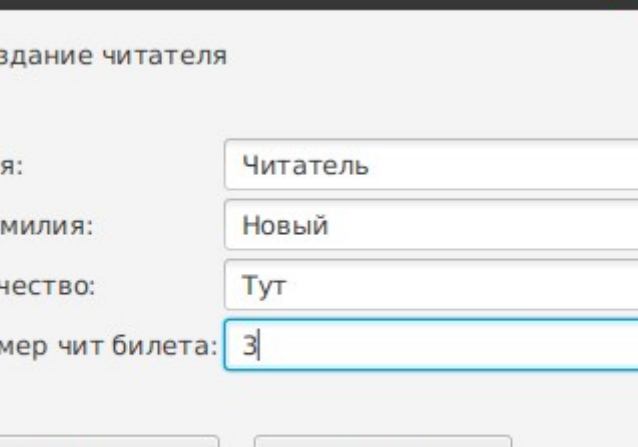
```

Тестирование

Отображение списка читателей и их чит. билета

[illegible]

Добавление читателя



Создание читателя

Имя: Читатель

Фамилия: Новый

Отчество: Тут

Номер чит билета: 3

Создать Отмена

Lab 9 Hauryliuk

Читатель	Читательский билет
Гаврилюк Рената Игоревна	1
Дубина Кристина Игоревна	2
Новый Читатель Тут	3

Читатель

Имя:

Читатель

Фамилия:

Новый

Отчество:

Тут

Номер ЧБ:

3

Открыть ЧБ

Редактирование читателя:

Lab 9 Nauryliuk

Читатель	Читательский билет
Гаврилюк Рената Игоревна	1
Дубина Кристина Игоревна	2
Новый Читатель Тут	3

Редактировать читателя

Имя:

Читатель

Фамилия:

Новый

Отчество:

Редактируемый

Номер чит билета:

3

Редактировать

Отмена

Читатель	Читательский билет
Гаврилюк Рената Игоревна	1
Дубина Кристина Игоревна	2
Новый Читатель Редактируемый	3

Читатель

Имя: Читатель

Фамилия: Новый

Отчество: Редактируемый

Номер ЧБ: 3

Открыть ЧБ

Редактировать

Поиск по читателю:

Lab 9 Hauryliuk

Читатель	Читательский билет
Дубина Кристина Игоревна	2

Читатель

Имя:

Фамилия:

Отчество:

Номер ЧБ:

Открыть ЧБ

Редактировать

Дубина

Найти

Удаление читателя (результат удаления читателя Новый Читатель
Редактированный):

Lab 9 Hauryliuk		
Читатель	Читательский билет	Читатель
Гаврилюк Рената Игоревна	1	Имя:
Дубина Кристина Игоревна	2	Фамилия:
		Отчество:
		Номер ЧБ:

Сортировка читателей по дате добавления:

Lab 9 Hauryliuk		
Читатель ▲	Читательский билет	
Гаврилюк Рената Игоревна	1	
Дубина Кристина Игоревна	2	

Lab 9 Hauryliuk		
Читатель ▼	Читательский билет	
Дубина Кристина Игоревна	2	
Гаврилюк Рената Игоревна	1	

Читательский билет:

[illegible]

Добавление записи (с созданием новой книги и автора):

The screenshot displays three overlapping windows from a library management application:

- Top Window: "Создание книги" (Create Book)**
 - Fields: "Название:" (Name), "Автор:" (Author), "Дата издания:" (Date of publication).
- Middle Window: "Добавление книги" (Add Book)**
 - Fields: "Книга:" (Book), "Дата выдачи:" (Issue date), "Дата сдачи:" (Due date).
 - Button: "Добавить" (Add).
- Bottom Window: "Создание автора" (Create Author)**
 - Fields: "Имя:" (Name) with value "Автор", "Фамилия:" (Surname) with value "Новый", "Отчество:" (Patronymic) with value "Книгр".
 - Buttons: "Создать" (Create), "Отмена" (Cancel).

Добавление книги в читательский билет

Книга:

Создание книги

Название:

Автор:

Дата издания:

Кол-во страниц:

Добавление книги в читательский билет

Книга:

Дата выдачи:

Дата сдачи:

Книга	Дата выдачи	Дата сдачи	Читательский билет	
Jupyter	2021-02-04	2021-02-19	Номер:	1
Python	2021-02-02	2021-02-28	Имя:	Рената
C++	2021-01-01	2021-02-25	Фамилия:	Гаврилюк
Название	2021-02-06	2021-02-28	Отчество:	Игоревна
			<input type="button" value="Редактировать читателя"/>	
			Книга	
			Название	Название
			Автор:	НовыйАвторКнигр
			Кол-во Стр.:	2001
			Год издания:	293
			Дата выдачи:	2021-02-06
			Дата сдачи:	2021-02-28
			<input type="button" value="Редактировать книгу"/>	
			<input type="button" value="Редактировать запись"/>	

Редактирование записи:

Книга	Дата выдачи	Дата сдачи
Jupyter	2021-02-04	2021-02-19
Python	2021-02-02	2021-02-28
C++	2021-01-01	2021-02-25

Редактировать книгу

Книга:

Дата выдачи:

Дата сдачи:

Книга	Дата выдачи	Дата сдачи
Jupyter	2021-02-04	2021-02-19
Python	2021-02-02	2021-02-28
C++	2021-01-01	2021-02-25
Название	2020-02-01	2021-02-28

Удаление записи (последней добавленной)

Книга	Дата выдачи	Дата сдачи
Jupyter	2021-02-04	2021-02-19
Python	2021-02-02	2021-02-28
C++	2021-01-01	2021-02-25

Вывод: приобрела практические навыки разработки многооконных приложений на JavaFX для работы с базами данных.