



Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего
профессионального образования
«Санкт-Петербургский государственный электротехнический университет
“ЛЭТИ” им.В.И.Ульянова (Ленина)»

Кафедра ВТ

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К КУРСОВОМУ ПРОЕКТУ
по дисциплине «ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ
ПРОГРАММИРОВАНИЕ»

**«Создание программного комплекса средствами объектно-
ориентированного программирования»**

Выполнил студент Побережный Е.С..
Факультет КТИ
Группа № 3312

Руководитель Павловский М.Г.

Подпись преподавателя _____

Санкт-Петербург
г 2024 г

1. Техническое задание

Введение

Программный комплекс (ПК) «Почтовая информационная система» разработан для автоматизации работы с информацией о клиентах, почтальных, газетах и журналах. ПК предоставит удобный инструмент для учета данных, связанных с подписками и распределением периодических изданий.

Основание для разработки

Разработка ПК «Почтовая информационная система» выполняется в рамках курсового проекта по дисциплине «Объектно-ориентированное программирование».

Назначение разработки

ПК «Почтовая информационная система» создан для оптимизации процессов, связанных с работой почты, включая учет информации о клиентах, почтальных и периодике. Основные задачи комплекса:

Ведение учета клиентов, подписавшихся на определенные газеты или журналы.

Систематизация данных о поступлении и распределении газет и журналов.

Управление сведениями о количестве экземпляров изданий.

Обеспечение удобного интерфейса для добавления, редактирования и удаления информации.

Требования к программе

Требования к функциональным характеристикам

Перечень функций

ПК должен поддерживать выполнение следующих операций:

Управление данными: добавление, изменение и удаление информации о клиентах, почтальных и изданиях.

Требования к составу выполняемых функций

Функция «Управление данными»

Ввод, просмотр, добавление, удаление и изменение информации должны поддерживать:

- данные о преподавателях (ФИО, классы, предмет);
- данные о учениках (ФИО, класс, успеваемости);

Требования к организации и форме представления выходных данных

Выходные данные должны отображаться в виде таблиц или списков.

Примеры представления:

Сводные таблицы с информацией о клиентах, почтальонах и газетах.

Списки или таблицы подписчиков конкретных газет или журналов.

Списки или таблицы почтальонов и их рабочих адресов.

Требования к организации и форме представления входных данных

Входные данные для ПК «Почтовая информационная система» содержат информацию о клиентах, почтальонах и периодических изданиях (газетах и журналах). Ввод данных осуществляется пользователем через интерфейс программы в режиме диалога. Для хранения данных используется XML-файл, в котором структурированная информация сохраняется в виде элементов и атрибутов.

Требования к надежности

Программный комплекс должен обеспечивать стабильную работу с XML-файлами. Основные аспекты:

Проверка целостности XML-файлов перед их использованием.

Использование обработки исключений для управления ошибками при чтении или записи данных (например, отсутствие файла, неверный формат XML).

Возможность восстановления данных из резервной копии XML-файла.

Условия эксплуатации

Программа предназначена для работы в режиме одного пользователя с монопольным доступом к XML-файлу. Это исключает конфликты при одновременном доступе к данным.

Требования к составу и параметрам технических средств

Для работы ПК «Почтовая информационная система» необходимы минимальные технические характеристики:

Процессор: Pentium III 500 МГц или выше.

Оперативная память: не менее 128 МБ.

Жесткий диск: от 4 ГБ свободного пространства.

Монитор: поддержка разрешения SVGA.

Устройства ввода: стандартная клавиатура и мышь.

Требования к информационной и программной совместимости

Программа должна быть совместима с операционной системой Windows и разработана на языке Java. Реализация должна включать:

- Принципы инкапсуляции (закрытые и открытые члены классов).
- Наследование для создания иерархий объектов.
- Конструкторы с параметрами.
- Абстрактные классы для определения базовых характеристик объектов.
- Обработка исключений для устойчивости работы.

Требования к программной документации

Программная документация должна соответствовать стандартам ЕСПД и включать:

- Подробное описание проектирования системы с использованием XML-файлов.
- Руководство пользователя с инструкциями по работе с программой.
- Полный комплект исходных текстов программы с комментариями.

Стадии и этапы разработки

Разработка технического задания

- Определение целей и задач программного комплекса.
- Формулировка функциональных и технических требований.
- Согласование структуры хранения данных в XML-файлах.

Описание вариантов использования ПК

- Формирование сценариев работы пользователя с системой, включая добавление, удаление и поиск данных.
- Подготовка примеров взаимодействия с XML-файлом для выполнения запросов, таких как получение списка подписчиков газеты.

Создание прототипа интерфейса пользователя

- Разработка эскизов графического интерфейса с основными элементами управления.
- Создание макета диалоговых окон для ввода и редактирования данных.

Разработка объектной модели ПК

- Определение основных сущностей (клиенты, почтальоны, издания).
- Построение иерархии классов для представления данных и взаимодействия с XML-файлом.

Построение диаграмм классов

- Определение классов, их атрибутов и методов.
- Построение связей между классами, таких как ассоциации, наследование и композиция.

Описание поведения ПК

- Моделирование сценариев использования программы.
- Определение последовательности действий для выполнения операций, таких как добавление нового клиента или формирования отчета.

Построение диаграмм действий

Создание диаграмм действий для визуализации выполнения ключевых процессов в системе.

Описание потоков данных и логики их обработки.

Порядок контроля и приемки

Приемка программного комплекса проводится в соответствии с заявленными требованиями технического задания.

2 Проектирование ПК

Описание вариантов использования ПК

Функциональность системы включает:

1. Управление клиентами (добавление, редактирование, удаление данных о клиентах).
2. Управление почтальонами (добавление, редактирование, удаление информации).
3. Ведение списка газет (категорий, тиражей, связанных клиентов).
4. Связь клиентов с выписываемыми газетами.
5. Привязка почтальонов к их рабочим адресам.

Актеры системы

1. **Администратор:**
 - Основной пользователь системы, выполняющий все операции (добавление, редактирование, удаление данных).
 - Иницирует загрузку и сохранение данных в XML-файл.
2. **Программист-разработчик:**
 - Использует систему для демонстрации её функционала.

Прецеденты использования

Основные сценарии взаимодействия с системой:

1. **Добавить данные:**
 - Клиента.
 - Почтальона.
 - Газету.
2. **Редактировать данные:**
 - Обновление информации о клиенте, почтальоне или газете.
3. **Удалить данные:**
 - Удаление записей из таблиц.
4. **Связать объекты:**
 - Привязка клиента к выписываемым газетам.
 - Привязка почтальона к обслуживаемым адресам.
5. **Работа с файлами:**
 - Загрузка данных из XML.
 - Сохранение изменений в XML.

Диаграмма прецедентов представлена на рис. 2.1.

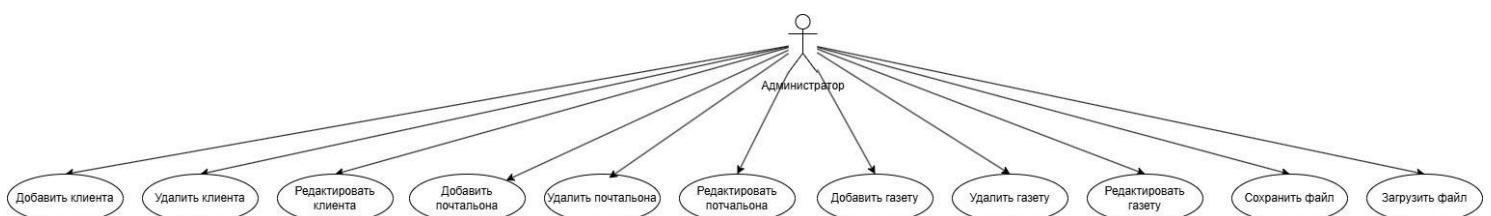


Рис.2.1 Диаграмма прецедентов

2.3 Создание прототипа интерфейса пользователя

Для детального описания интерфейса почтовой информационной системы все процессы взаимодействия с системой разбиваются на экранные формы. Каждая форма описывает элементы управления, которые позволяют пользователю вводить или изменять данные, а также отклики системы на действия пользователя.

Главная экранная форма представляет собой главное окно системы, в котором размещены меню с пунктами «Загрузить» и «Сохранить», панель с кнопками для действий «Добавить», «Редактировать» и «Удалить», а также комбинированный список для выбора таблицы (например, с клиентами, почтальонами, газетами и т. д.). При выборе пункта меню для загрузки или сохранения данных открывается соответствующее диалоговое окно. Когда пользователь выбирает таблицу, отображается информация, соответствующая выбранной категории, а при нажатии одной из кнопок («Добавить», «Редактировать», «Удалить») открывается окно для ввода или изменения данных.

Форма добавления или редактирования клиента позволяет ввести ФИО и адрес клиента. Форма удаления, удаляет поле клиента и его адрес. После ввода данных пользователь может нажать кнопку «Сохранить» для сохранения изменений или кнопку «Отмена» для отмены действия. При сохранении введенные данные сохраняются в таблице, а при отмене окно закрывается без изменений.

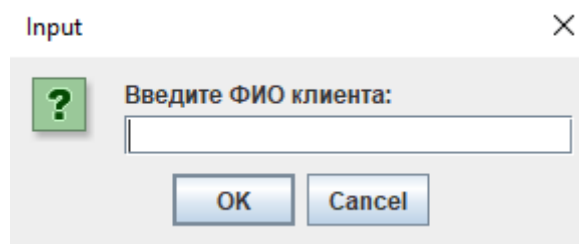
Аналогично работает форма добавления редактирования и удаления почтальона, где пользователь вводит ФИО почтальона и его график работы или поле для его удаления. Для сохранения изменений используется кнопка «Сохранить», а для отмены — кнопка «Отмена».

Форма добавления газеты включает поля для ввода названия газеты, тиража и категории. Данные сохраняются при нажатии кнопки «Сохранить», а отмена действия закрывает окно без изменений.

The screenshot shows a window titled "Почтовая информационная система" with standard Windows window controls. Below the title bar is a "Меню" (Menu) bar. Underneath is a "Выбор таблицы" (Select table) section with a dropdown menu currently set to "Клиенты". Below this is a table with two columns: "ФИО клиента" (Client's full name) and "Адрес" (Address). The table body is currently empty. At the bottom of the window are three buttons: "Добавить" (Add), "Редактировать" (Edit), and "Удалить" (Delete).

ФИО клиента	Адрес
-------------	-------

Рис. 2.2. Таблица клиентов



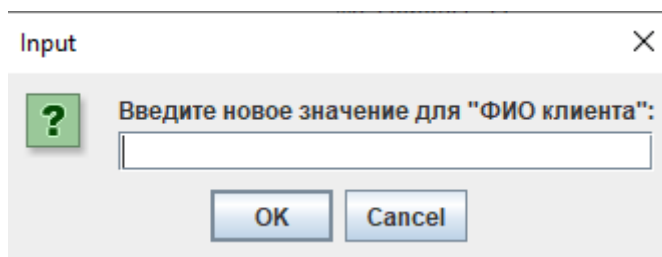
Input

Введите ФИО клиента:

OK Cancel

This is a standard Windows-style input dialog box. It has a title bar with the word "Input" and a close button (X). The main area contains a green square icon with a white question mark, followed by the text "Введите ФИО клиента:". Below this is a single-line text input field. At the bottom are two buttons labeled "OK" and "Cancel".

Рис. 2.3. Форма добавления клиента



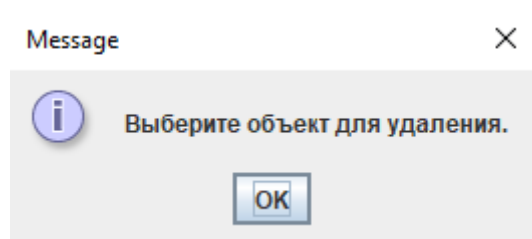
Input

Введите новое значение для "ФИО клиента":

OK Cancel

This is a standard Windows-style input dialog box, similar to the previous one. It has a title bar with the word "Input" and a close button (X). The main area contains a green square icon with a white question mark, followed by the text "Введите новое значение для 'ФИО клиента':". Below this is a single-line text input field. At the bottom are two buttons labeled "OK" and "Cancel".

Рис. 2.4. Форма редактирования таблицы клиентов



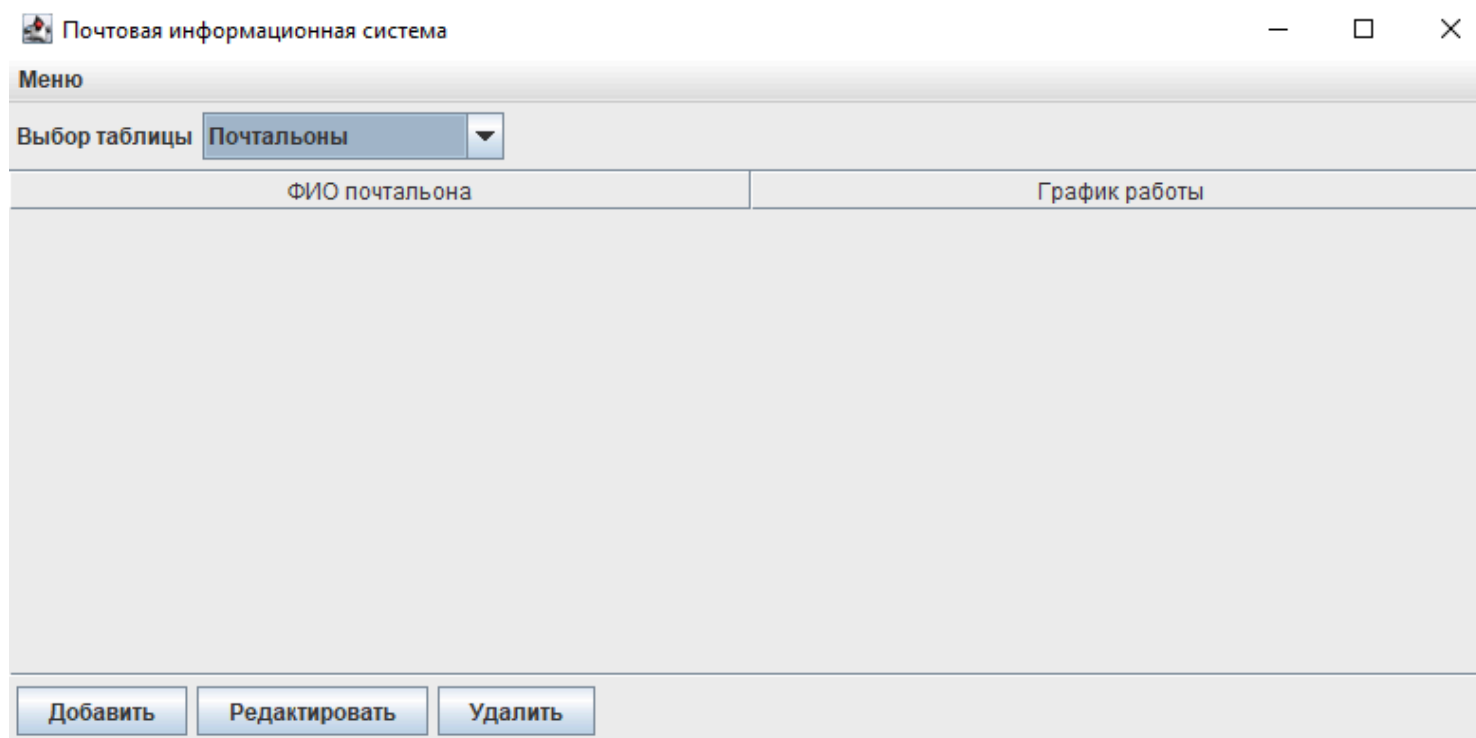
Message

Выберите объект для удаления.

OK

This is a standard Windows-style message dialog box. It has a title bar with the word "Message" and a close button (X). The main area contains a blue circular icon with a white lowercase 'i', followed by the text "Выберите объект для удаления.". Below this is a single button labeled "OK".

Рис. 2.5. Форма удаления клиентов



Почтовая информационная система

Меню

Выбор таблицы Почтальоны

ФИО почтальона	График работы
----------------	---------------

Добавить Редактировать Удалить

This is the main application window. The title bar shows the application name "Почтовая информационная система" and standard window controls (minimize, maximize, close). Below the title bar is a menu bar with a single item "Меню". Under the menu bar is a section labeled "Выбор таблицы" with a dropdown menu currently showing "Почтальоны". The main area of the window is a table with two columns: "ФИО почтальона" and "График работы". The table is currently empty. At the bottom of the window is a toolbar with three buttons: "Добавить", "Редактировать", and "Удалить".

Рис. 2.6. Таблица почтальонов

Input

Введите ФИО почтальона:

OK Cancel

Рис. 2.7. Форма добавления почтальона

Input

Введите новое значение для "ФИО почтальона":

OK Cancel

Рис. 2.8. Форма редактирования таблицы почтальоны

Message

Выберите объект для удаления.

OK

Рис. 2.9. Форма удаления почтальонов

Почтовая информационная система

Меню

Выбор таблицы: Газеты

Название газеты	Тираж	Категория

Добавить Редактировать Удалить

Рис. 2.10. Таблица газет

Input

Введите название газеты:

OK Cancel

Рис. 2.11. Форма добавления газеты

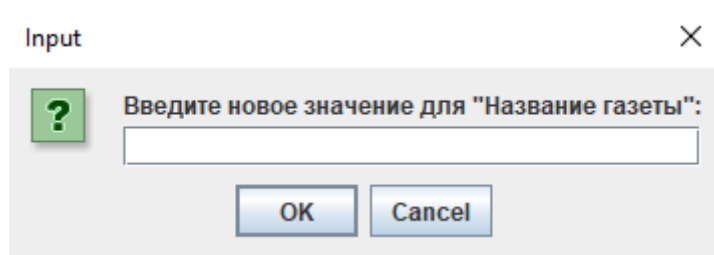


Рис. 2.12. Форма редактирования таблицы газеты

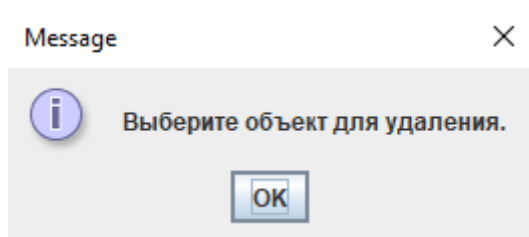


Рис. 2.13. Форма удаления газет

Таблица 2.1

Экранная форма	Элементы управления	Действия пользователя	Отклик системы
Регистрация клиента	Кнопки: «Добавить клиента», «Удалить клиента», «Редактировать клиента»	Нажать на кнопку «Добавить клиента», ввести данные в поля «ФИО клиента», «Адрес».	Добавление клиента в XML файл.
	Поля ввода: «ФИО клиента», «Адрес»	Нажать на кнопку «Удалить клиента», выбрать клиента для удаления.	Удаление клиента из XML файла.
		Нажать на кнопку «Редактировать клиента», изменить данные клиента.	Редактирование данных клиента в XML файле.
Регистрация почтальона	Кнопки: «Добавить почтальона», «Удалить почтальона», «Редактировать почтальона»	Нажать на кнопку «Добавить почтальона», ввести данные в поля «ФИО почтальона», «График работы».	Добавление почтальона в XML файл.
	Поля ввода: «ФИО почтальона», «График работы»	Нажать на кнопку «Удалить почтальона», выбрать почтальона для удаления.	Удаление почтальона из XML файла.
		Нажать на кнопку «Редактировать почтальона», изменить данные почтальона.	Редактирование данных почтальона в XML файле.
Работа с газетами	Кнопки: «Добавить газету», «Удалить газету», «Редактировать газету»	Нажать на кнопку «Добавить газету», ввести данные в поля «Название газеты», «Тираж», «Категория».	Добавление газеты в XML файл.
	Поля ввода: «Название газеты», «Тираж», «Категория»	Нажать на кнопку «Удалить газету», выбрать газету для удаления.	Удаление газеты из XML файла.

		Нажать на кнопку «Редактировать газету», изменить данные о газете.	Редактирование данных о газете в XML файле.
Сохранение данных	Кнопки: «Сохранить данные»	Нажать на кнопку «Сохранить данные», внести изменения в нужные поля.	Сохранение изменений в XML файле.
Загрузка данных	Кнопки: «Загрузить данные»	Нажать на кнопку «Загрузить данные» для загрузки данных из XML файла.	Загрузка данных из XML файла.

2.3. Разработка объектной модели ПК

Объектная модель программы для управления почтовой информационной системой отражает основные концепции предметной области в виде набора сущностей, их атрибутов и операций. В данной реализации сущности представлены в виде классов, связанных с функциональностью работы с клиентами, почтальонами, газетами и их связями. В процессе проектирования были выделены следующие сущности: Клиент, Почтальон, Газета, а также их связи. Клиент характеризуется такими атрибутами, как имя клиента (строка) и адрес клиента (строка). Для клиента предусмотрены операции добавления нового клиента, редактирования существующего клиента и удаления клиента. Почтальон имеет атрибуты имени почтальона (строка) и графика работы (строка) и аналогично поддерживает операции добавления, редактирования и удаления. Газета содержит атрибуты названия газеты (строка), тиража (целое число) и категории (строка), для нее также реализованы операции добавления, редактирования и удаления.

Связи между сущностями представлены классами ClientNewspaperRelation и PostmanAddressRelation. Связь между клиентом и газетой включает атрибуты клиента и списка газет, а связь между почтальоном и адресами – атрибуты почтальона и списка адресов. Важной особенностью реализации является автоматическое формирование этих связей на основании данных из таблиц «Клиенты», «Газеты» и «Почтальоны». Это означает, что пользователь не может вручную добавлять или удалять связи в соответствующих таблицах, так как они обновляются автоматически при изменении данных в исходных таблицах. Например, связь между клиентами и газетами обновляется при изменении данных в таблицах «Клиенты» или «Газеты», что позволяет поддерживать консистентность данных в программе.

Ассоциации между сущностями отображают логические связи, такие как подписка клиента на газету или обслуживание адресов почтальоном. Эти ассоциации формируются через списки, связывающие объекты разных типов. Программа использует графический интерфейс на основе Swing, который включает таблицы для отображения данных о клиентах, почтальонах, газетах и их связях, меню для загрузки данных из XML-файла и сохранения изменений, а также панель управления с кнопками для добавления, редактирования и удаления записей. Все данные хранятся в виде объектов классов, а для их сохранения и загрузки используется формат XML, что позволяет удобно работать с информацией и сохранять изменения для последующего использования.

Детальное описание операций представлено в табл. 2.2.

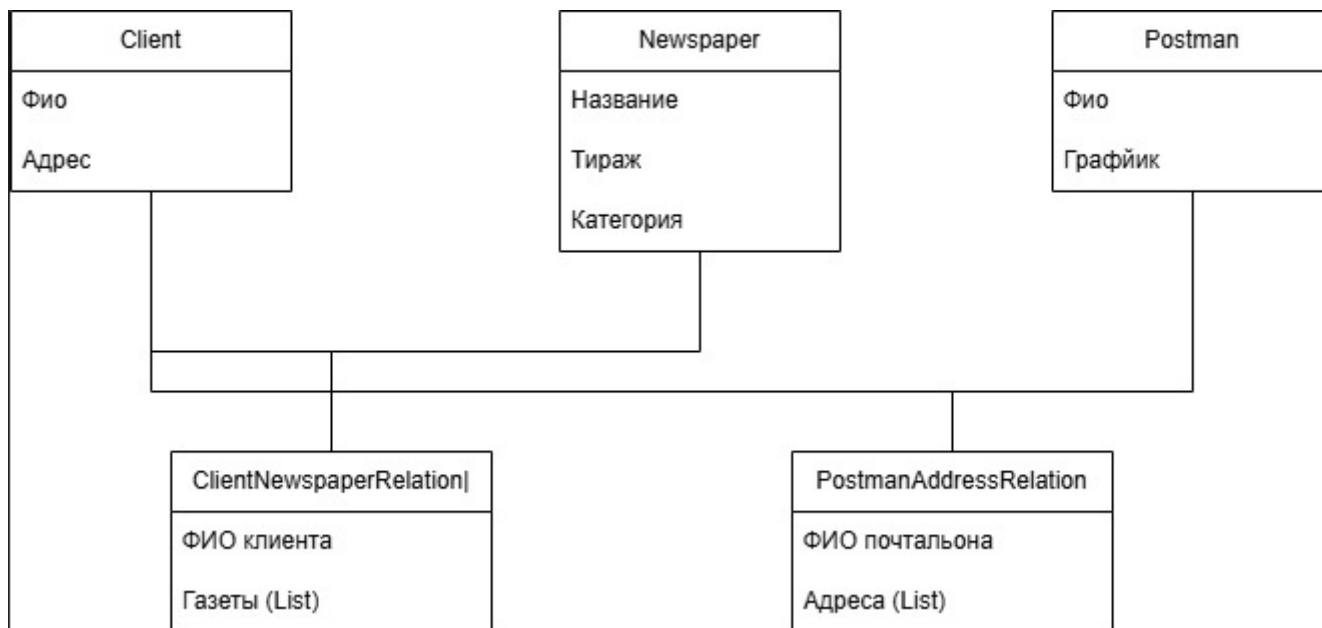


Рис. 2.14. Диаграмма сущностей

Построение диаграммы программных классов

Диаграмма классов (class diagram) используется для отображения программных классов и их взаимодействий, которые будут применяться в системе, например, для управления почтовыми сервисами. Диаграмма основывается на объектной модели и описывает три ключевых элемента каждого класса: имя класса, его атрибуты и методы.

Структура классов:

1. Имя класса — это название сущности (например, Postman, Client, Newspaper), или оно может отражать специфику программной реализации.
2. Атрибуты класса — описывают характеристики сущности (например, id: int, name: String, workSchedule: String).
3. Методы класса — описывают действия, которые класс может выполнять (например, getName(): String, getId(): int).

Каждый класс изображается в виде прямоугольника, разделённого на три секции:

1. **Имя класса** — на первой секции.
2. **Атрибуты** — на второй секции, где после двоеточия указывается тип данных.
3. **Методы** — на третьей секции, где указывается тип возвращаемого значения.

Права доступа:

Атрибуты и методы классов помечаются модификаторами доступа:

1. + (public) — открытый доступ, доступен извне.
2. - (private) — закрытый доступ, доступен только внутри класса.
3. # (protected) — доступ для наследования, доступен внутри класса и его подклассов.

Описание атрибутов и методов:

1. Для атрибутов указывается тип данных после двоеточия, например, name: String, id: int.
2. Для методов указывается тип возвращаемого значения после скобок, например, getName(): String, getId(): int. Конструкторы не имеют возвращаемого значения, и они в диаграмме не помечаются типом данных.

Типы связей между классами:

1. Ассоциация — связь между классами, которая может быть однонаправленной или двунаправленной. В однонаправленной ассоциации стрелка указывает на класс, который инициирует запрос к другому классу.

Пример: Связь между классами Postman и PostmanAddressRelation — один почтальон может обслуживать несколько клиентов, и связь между ними реализуется через ассоциацию.

2. Агрегирование — отношение «целое/часть», где один объект является частью другого. Например, класс ClientNewspaperRelation может быть частью класса Client, где один клиент может подписываться на несколько газет.

Агрегирование изображается полым ромбом, указывающим на агрегирующий класс. В сильном агрегировании, когда часть не существует без целого, ромб заполняется.

3. Наследование — отношение «общее-частное», когда один класс наследует свойства и методы другого. Это изображается стрелкой с полым треугольником, направленной от производного класса к базовому.

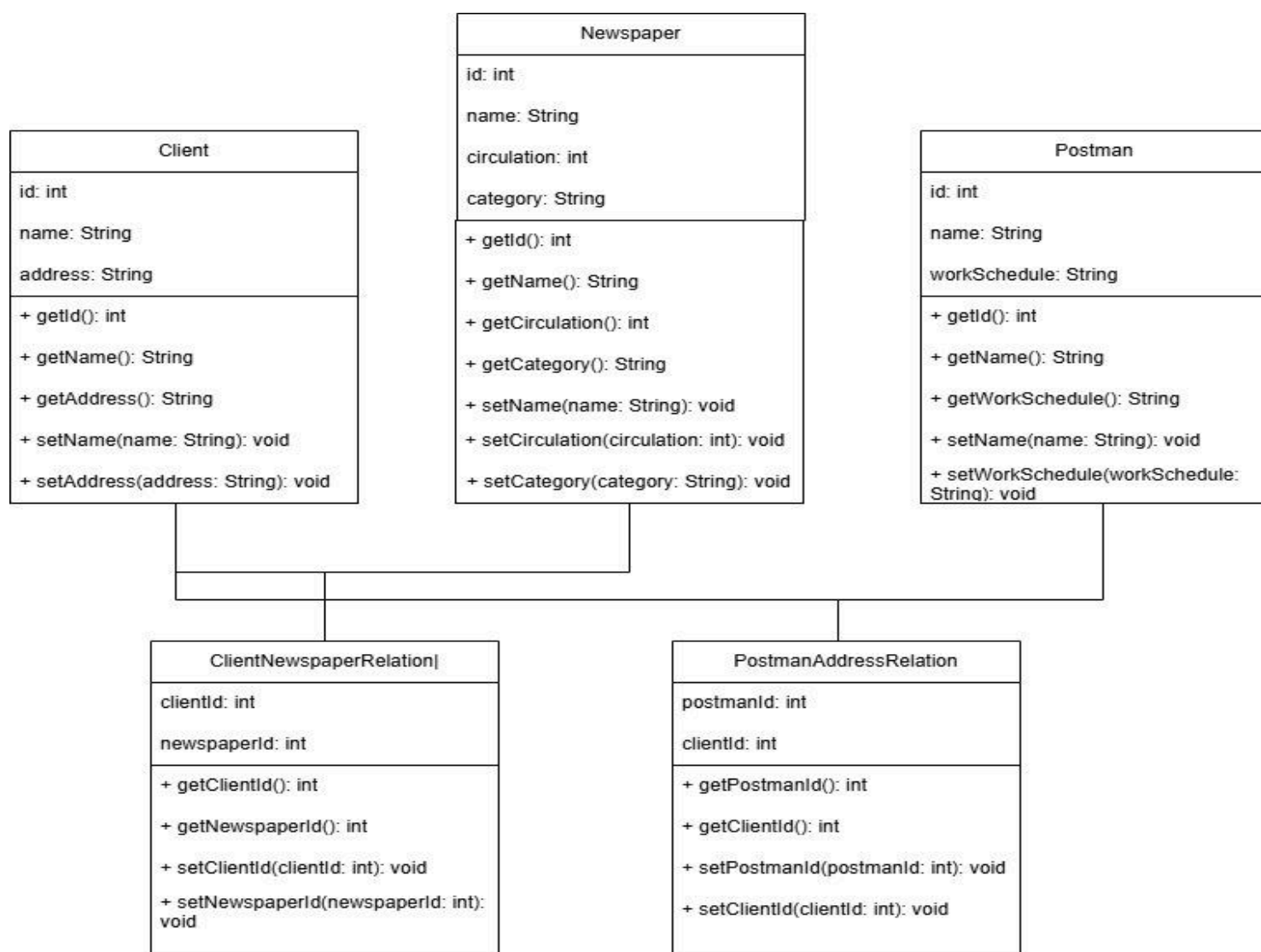


Рис.2.15. Диаграмма классов

2.4 Описание поведения программного комплекса для вашего кода

Описание поведения программного комплекса отражает последовательность действий, которые выполняются системой в ответ на запросы пользователей или других объектов. Это описание не включает технические детали реализации, а лишь отображает порядок взаимодействия объектов, что особенно важно для анализа работы системы на уровне сценариев. Для такого описания используется диаграмма последовательностей (Sequence Diagram).

Диаграмма последовательностей представляет собой схему, которая показывает взаимодействие между пользователями и объектами системы в рамках выбранного сценария. Она отображает порядок вызовов операций и последовательность их выполнения.

Основные элементы диаграммы последовательностей:

Вертикальная ось — представляет собой временную шкалу, на которой отображается последовательность выполнения операций.

Горизонтальная ось — отображает участников взаимодействия, которые представлены объектами и пользователями системы.

Для построения диаграммы последовательностей для вашего кода, можно выделить несколько шагов:

1. Идентификация участников взаимодействия

Необходимо определить пользователей и объекты программных классов, которые будут участвовать в конкретном сценарии. Например, для вашего кода это могут быть объекты классов `Postman`, `Client`, `Newspaper`, `ClientNewspaperRelation`, и `PostmanAddressRelation`. Эти объекты изображаются в верхней части диаграммы в виде прямоугольников в одну линию. В каждом прямоугольнике указывается:

Имя объекта (подчеркнутое).

Название класса, к которому принадлежит объект.

2. Определение линий жизни объектов

Для каждого участника взаимодействия на диаграмме рисуется вертикальная пунктирная линия, называемая линией жизни. Линия жизни отображает продолжительность существования объекта или пользователя в рамках сценария.

3. Выбор операций из объектной модели

Для реализации сценария необходимо определить, какие операции из диаграммы классов будут использованы. Например, из вашего кода могут быть вызваны методы таких классов как `getId()`, `setName()`, `getClientId()`, и другие. Если какие-то операции не были описаны в модели, их следует добавить.

4. Отображение запросов и вызовов операций

Запросы и вызовы операций отображаются на диаграмме с помощью горизонтальных стрелок:

Начало стрелки — вертикальная линия объекта или пользователя, инициирующего операцию.

Конец стрелки — вертикальная линия объекта, который выполняет операцию.

Над стрелкой указывается:

Номер операции.

Имя операции.

Параметры операции (в скобках).

При необходимости, добавляется комментарий (начинается с //).

5. Обозначение выполнения операций

Выполнение операций на диаграмме изображается прямоугольниками, которые размещаются на линии жизни объекта. Порядок выполнения операций определяется:

Нумерацией операций, которая ставится перед именем операции (например, 1, 2, 3).

Положением горизонтальной стрелки на диаграмме: чем ниже стрелка, тем позже операция будет выполняться.

6. Вложенная нумерация

Если одна операция вызывает другие операции, внутри нее используется вложенная система нумерации (например, 1, 1.1, 1.1.1). Нумерация на каждом уровне вложенности начинается с 1.

7. Условные конструкции и жизненный цикл объектов

Условные операции (if-else) могут быть отображены с помощью ветвлений, которые показывают различные пути выполнения в зависимости от условий.

Создание и уничтожение объектов:

Если объект создается в процессе сценария, его линия жизни начинается с точки, где объект создается.

Уничтожение объекта отображается на диаграмме с помощью символа "X" на линии жизни объекта.

8. Связь многие ко многим в системе управления почтовыми данными

Связь многие ко многим реализована между различными сущностями в системе управления почтовыми данными. В качестве объектов, между которыми установлена такая связь, используются следующие сущности: клиенты, почтальоны и газеты. Эта связь позволяет эффективно моделировать и управлять взаимоотношениями между сущностями, сохраняя их независимость и минимизируя дублирование данных.

1. Связь между клиентами и газетами

В рамках системы реализована связь многие ко многим между сущностями "Клиент" и "Газета". Каждый клиент может подписываться на несколько газет, и каждая газета может быть подписана несколькими клиентами. Для реализации этой связи в системе используется отдельная таблица ClientNewspaperRelation, которая содержит два ключевых столбца: clientId и newspaperId. Эти столбцы представляют собой идентификаторы клиента и газеты, создавая таким образом запись о подписке клиента на газету.

Пример:

Клиент А подписан на Газету X и Газету Y.

Клиент В подписан только на Газету X.

В таблице ClientNewspaperRelation для этих записей будут следующие данные:

Для клиента А: записи с clientId для клиента А и newspaperId для Газеты X и Газеты Y.

Для клиента В: запись с clientId для клиента В и newspaperId для Газеты X.

Такой подход позволяет хранить все подписки в одной таблице и легко масштабировать систему при увеличении числа подписок.

2. Связь между почтальонами и клиентами

Для модели "Почтальон - Клиент" также реализована связь многие ко многим, поскольку один почтальон может обслуживать несколько клиентов, а каждый клиент может быть обслужен несколькими почтальонами. Для этой связи используется таблица PostmanAddressRelation, которая содержит два ключевых столбца: postmanId и clientId. Эти столбцы ссылаются на идентификаторы почтальона и клиента, создавая запись о том, что определенный почтальон обслуживает определенного клиента.

Пример:

Почтальон 1 обслуживает клиентов А, В и С.

Почтальон 2 обслуживает только клиента В.

В таблице PostmanAddressRelation будут следующие записи:

Для почтальона 1: записи с postmanId для почтальона 1 и clientId для клиентов А, В и С.

Для почтальона 2: запись с postmanId для почтальона 2 и clientId для клиента В

3. Руководство оператора

Назначение программы

Программа предназначена для автоматизации процессов управления почтовой системой. Она позволяет администратору системы эффективно управлять данными о клиентах, почтальонах, газетах и их взаимосвязях. В рамках программы администратор может:

- Добавлять, изменять и удалять информацию о клиентах.
- Добавлять, изменять и удалять информацию о почтальонах.
- Добавлять, изменять и удалять информацию о газетах.
- Управлять связями между клиентами и газетами.
- Управлять связями между почтальонами и клиентами.
- Загрузка и сохранение данных в формате XML.
- Просматривать и обновлять информацию в таблицах для удобного анализа и управления.

Условия выполнения программы

Программа предназначена для работы в операционной системе Windows (7 и выше) с установленной поддержкой Java. Для корректной работы программы требуется наличие Java Development Kit (JDK) версии 8 или выше.

Минимальные характеристики:

- **Операционная система:** Windows 7 и выше.
- **Процессор:** с тактовой частотой от 1.8 ГГц.
- **Оперативная память:** не менее 4 Гб.
- **Жесткий диск:** не менее 500 Мб свободного места.
- **Программное обеспечение:** Java JDK 8 или выше.

Описание задачи

Программа предоставляет интерфейс для управления данными почтовой системы, включая информацию о клиентах, почтальонах и газетах. Администратор может обновлять данные о клиентах и почтальонах, а также их взаимосвязи с газетами и почтовыми адресами. Кроме того, программа поддерживает загрузку и сохранение данных в формате XML, что позволяет удобно работать с данными и интегрировать программу с другими системами.

Основные сущности программы:

- **Клиенты:** информация о клиентах, включая имя и адрес.
- **Почтальоны:** информация о почтальонах, включая их расписание работы.
- **Газеты:** информация о газетах, включая название, тираж и категорию.
- **Связи между клиентами и газетами:** связь, указывающая, какие газеты доставляются каким клиентам.
- **Связи между почтальонами и клиентами:** связь, указывающая, какой почтальон обслуживает каких клиентов.

Входные и выходные данные

Входные данные:

- Информация о клиентах: имя, адрес.
- Информация о почтальонах: имя, расписание работы.
- Информация о газетах: название, тираж, категория.
- Связи между клиентами и газетами, почтальонами и клиентами.

Выходные данные:

- Отчеты по клиентам, почтальонам и газетам.

- Информация о взаимосвязях клиентов и газет, почтальонов и клиентов.
- Возможность загрузки и сохранения данных в XML.

Выполнение программы

Подготовка к запуску

Для корректной работы программы необходимо установить Java Development Kit (JDK) версии 8 или выше. Программа не требует установки дополнительного серверного ПО, так как использует файловое хранилище для данных.

Запуск программы

При запуске программы на экране появится главное окно, в котором будут отображаться таблицы с данными о клиентах, почтальонах и газетах.

Главное окно:

1. **Таблица клиентов** – отображает список всех клиентов с их именами и адресами.
2. **Таблица почтальонов** – отображает информацию о почтальонах с их расписаниями.
3. **Таблица газет** – отображает список газет с информацией о тираже и категории.
4. **Таблица связей клиентов и газет** – отображает, какие газеты доставляются каким клиентам.
5. **Таблица связей почтальонов и клиентов** – отображает, какой почтальон обслуживает какого клиента.

Основные функции программы

1. **Добавление и редактирование данных:**
 - Чтобы добавить клиента, почтальона или газету, нужно нажать кнопку "Добавить" и ввести необходимые данные в соответствующие поля.
 - Для редактирования данных выберите нужную строку в таблице и нажмите кнопку "Изменить".
2. **Удаление данных:**
 - Чтобы удалить клиента, почтальона или газету, выберите нужную строку в таблице и нажмите кнопку "Удалить".
3. **Загрузка данных:**
 - Для загрузки данных из файла XML нажмите кнопку "Загрузить данные". Программа откроет диалоговое окно выбора файла. После выбора файла данные будут загружены в соответствующие таблицы.
4. **Сохранение данных:**
 - Для сохранения данных в XML формат нажмите кнопку "Сохранить данные". Выберите путь и имя файла, куда будут сохранены данные.

Пример работы с данными

1. **Добавление нового клиента:**
 - Перейдите на вкладку "Клиенты".
 - Заполните поля для имени и адреса клиента.
 - Нажмите кнопку "Добавить клиента". Клиент появится в таблице.
2. **Добавление нового почтальона:**
 - Перейдите на вкладку "Почтальоны".
 - Заполните поля для имени и расписания работы почтальона.
 - Нажмите кнопку "Добавить почтальона". Почтальон появится в таблице.
3. **Добавление газеты:**
 - Перейдите на вкладку "Газеты".
 - Заполните поля для названия газеты, ее тиража и категории.
 - Нажмите кнопку "Добавить газету". Газета будет добавлена в таблицу.
4. **Добавление связи между клиентом и газетой:**
 - Перейдите на вкладку "Связи клиентов и газет".
 - Введите ID клиента и ID газеты.
 - Нажмите кнопку "Добавить связь".
5. **Добавление связи между почтальоном и клиентом:**
 - Перейдите на вкладку "Связи почтальонов и клиентов".
 - Введите ID почтальона и ID клиента.

- Нажмите кнопку "Добавить связь".

Использование операций с файлами

- **Загрузка данных:** Для загрузки данных из XML файла, нажмите кнопку "Загрузить данные". В появившемся диалоговом окне выберите XML файл для загрузки данных в программу.
- **Сохранение данных:** Для сохранения данных в XML файл, нажмите кнопку "Сохранить данные". В диалоговом окне выберите место и имя файла, в который нужно сохранить данные

Заключение

В ходе выполнения курсовой работы была разработана и реализована программа для администрирования почтовой системы, которая включает функциональность для управления данными о клиентах, почтальонах, газетах и их взаимосвязях. Программа предоставляет пользователю удобный интерфейс для работы с этими данными, включая возможность добавлять, изменять и удалять записи, а также управлять связями между различными сущностями системы.

Одной из ключевых особенностей программы является поддержка загрузки и сохранения данных в формате XML, что позволяет интегрировать систему с внешними источниками данных и обеспечивать удобную работу с файлами. Работа программы основана на использовании стандартных библиотек Java, что гарантирует ее совместимость с большинством современных операционных систем и удобство в эксплуатации.

В процессе разработки особое внимание было уделено удобству интерфейса и функциональной целесообразности предоставленных возможностей. Программа позволяет эффективно решать задачи, связанные с управлением почтовыми операциями, такими как организация доставки газет и управление почтальонами, а также хранение и обработка информации о клиентах.

Таким образом, выполненная работа позволила достичь поставленных целей, а разработанная программа представляет собой удобное и эффективное средство для администрирования почтовой системы, способное значительно упростить работу с данными и повысить производительность системы в целом. В дальнейшем можно рассмотреть расширение функционала программы, например, добавление возможностей для более сложной аналитики или реализации сетевого взаимодействия для работы с удаленными базами данных.

Ссылки:

<https://github.com/Burav7-8/OOP/tree/main/cw>