Федеральное государственное автономное

образовательное учреждение высшего образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий

Кафедра вычислительной техники

**КурсовАя работа**

Информационная система «Театр»

Руководитель \_\_\_\_\_\_\_\_\_ Сидоров А.Ю.

подпись, дата

Студент КИ19-06Б, 031940417 \_\_\_\_\_\_\_\_\_\_ Шнайдер А.В.

подпись, дата

Красноярск 2020

# Реферат

Темой курсовой работы является информационная система «Театр».

Курсовая работа состоит из:

16 страниц;

12 рисунков

Цель работы: создание корректно работающей информационной системы «Театр» и улучшение практических навыков работы с современными средами разработки.

Задачи работы:

1. Изучение объектно-ориентированных проектирования и программирования в рамках выполнения курсовой работы;

2. Создание корректно работающего функционального интерфейса информационной системы;

Информационная система «Театр» предназначена для управления информацией о представлениях, актёрах и проданных билетах, что существенно упростило бы работу сотрудникам театров.

ИНФОРМАЦИОННАЯ СИСТЕМА, ПРОГРАММА, ПОЛЬЗОВАТЕЛЬ, АДМИНИСТРАТОР, КАССИР, ЗРИТЕЛЬ, БИЛЕТ, ПРЕДСТАВЛЕНИЕ, АКТЁР.

Для создания итоговой программы с графическим интерфейсом была использована среда разработки QtCreator.

**СОДЕРЖАНИЕ**

1 Постановка задачи 4

2 Подходы к решению поставленной задачи 4

2.1. Описание входных и выходных данных 4

2.2.Требоавния к составу выполняемых функций и макет интерфейса 5

3 Список идентификаторов и их назначение 12

3.1. Связи классов 12

3.2. Список файлов и их предназначение 14

4 Подробное описание алгоритма 14

4.1. Обоснование выбора алгоритма 14

4.2. Описание алгоритма 15

5 Инструкция к программе 15

Заключение 15

Список использованных источников 16

Введение

Информационная система «Театр» предназначена для использования в театрах. Данная система предоставляет и редактирует информацию о представлениях, актёрах и купленных билетах. Таким образом, программа должна существенно упростить систематизацию информации театра.

# Постановка задачи

Информационная система «Театр». Информация: актеры – фамилия, имя, отчество, дата рождения, список спектаклей, в которых занят актер, роли, исполняемые в данных спектаклях; спектакли – автор пьесы, название, постановщик, список действующих лиц и исполнителей; афиша – название спектакля, дата, время; билеты – дата, количество билетов разного типа (партер, ложи, балкон и др.). Операции: коррекция списка актеров и списка спектаклей; составление и коррекция афиши; продажа билетов на спектакли; справки (участие актеров в различных постановках, информация по спектаклям и др.). Предусмотреть следующие группы пользователей: администратор, кассир, зритель.

# Подходы к решению поставленной задачи

**2.1. Описание входных и выходных данных**

Строка представляется типом QString.

При сохранении в двоичный файл класс QString помещает сначала длину строки в байтах (quint32), за которой следуют данные в UTF-16.

Число – 32-битное целое число со знаком.

В качестве входных и выходных данных выступают файлы:

users.bin – бинарный файл с данными о пассажирах:

- логин: строка;

- пароль: строка;

- уровень доступа: число от 1 до 3, где 1 – администратор, 2 – кассир, 3 – зритель.

viewer.bin – бинарный файл с данными о билетах зрителей, состоящий из строк вида:

- логин: строка;

- количество билетов: число;

- список билетов: название спектакля (строка), дата и время спектакля (строка вида чч:мм ДД.ММ.ГГГГ), количество купленных мест (число) и тип мест (строка).

performance.bin – бинарный файл с данными о спектаклях:

- автор пьесы: строка;

- название спектакля: строка;

- постановщик: строка;

- список ролей и актёров: строка вида «актёр - роль, актёр - роль …»;

- время проведения спектакля: строка чисел вида «мм:чч ДД.ММ.ГГГГ, мм:чч ДД.ММ.ГГГГ …».

actors.bin – бинарный файл с данными об актёрах:

- ФИО актёра – строка;

- дата рождения актёра – строка чисел вида ДД.ММ.ГГГГ;

- роли актёра – строка вида «название спектакля – роль, название спектакля – роль…».

**2.2.Требоавния к составу выполняемых функций и макет интерфейса**

На рисунке 1 представлена диаграмма прецендентов.

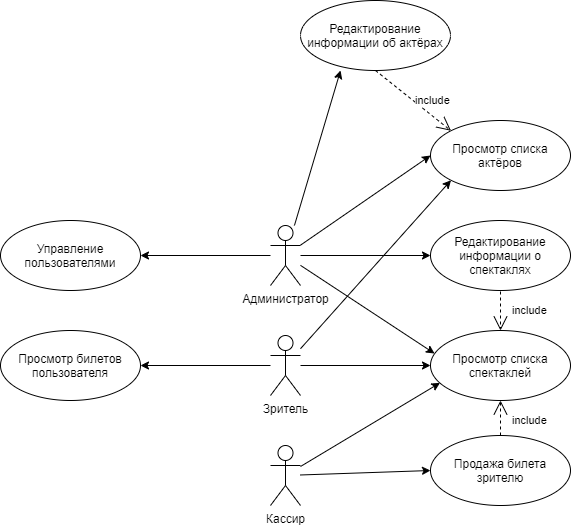


Рисунок 1 – Диаграмма прецедентов

Название прецедента: Управление пользователями

Действующее лицо: Администратор

Цель: Внести данные о новом пользователе в систему или удалить данные о пользователе

Предусловие: Наличие пользователя, которого необходимо зарегистрировать или удалить; администратор уже выполнил вход в систему

Главная последовательность:

1. При нажатии в главном окне (рисунок 2) кнопки «Список пользователей» открывается окно со списком пользователей системой (рисунок 3)



Рисунок 2 – Главное окно

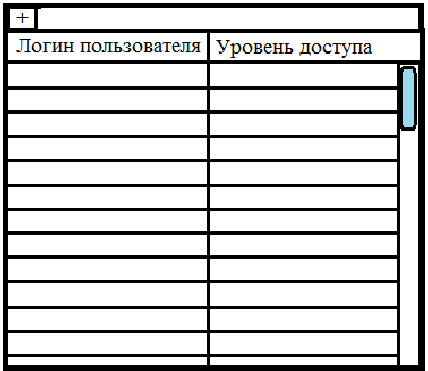


Рисунок 3 – Окно списка пользователей

2. Для добавления нового пользователя нужно нажать кнопку «+» в открывшемся окне. После этого откроется окно регистрации (рисунок 4).

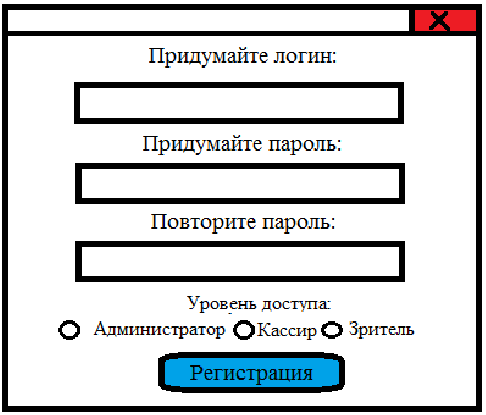


Рисунок 4 – Окно регистрации

3. Вводятся данные (логин и пароль, а также нужный уровень доступа).

4. После нажатия кнопки «Регистрация» проверяется правильность введённых данных. При неправильно введённых данных (уже существует пользователь с таким логином или поле пароля пусто) выводится окно с предупреждением об этом. Если же данные введены правильно, они вносятся в файл с логинами и паролями пользователей, а окно регистрации закрывается.

5. Для удаления пользователя в окне списка пользователей нужно дважды щёлкнуть левой кнопкой мыши по строке нужного пользователя. После этого откроется окно с подтверждением удаления.

6. Чтобы сохранить изменения, необходимо нажать кнопку сохранения в верхней части окна.

Название прецедента: Просмотр билетов пользователя

Действующее лицо: Зритель

Цель: Просмотреть билеты пользователя

Предусловие: Зритель выполнил вход в систему

Главная последовательность:

1. В главном окне при нажатии кнопки «Просмотреть список купленных билетов» открывается окно со списком билетов пользователя (рисунок 5).

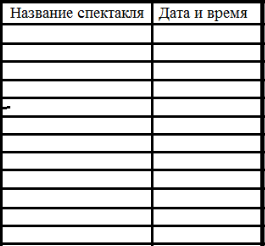


Рисунок 5 – Окно списка билетов

2. При двойном нажатии левой кнопкой мыши по интересующей строке с билетом будет открыто окно с подробной информацией о билете (рисунок 6).

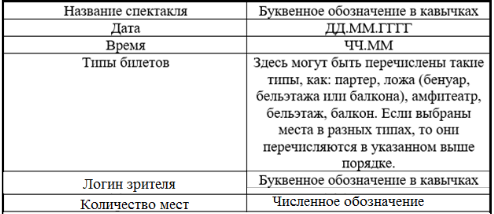


Рисунок 6 – Окно подробной информации о билете

Название прецедента: Просмотр списка спектаклей

Действующее лицо: Любой пользователь

Цель: Просмотреть список спектаклей

Предусловие: Пользователь должен выполнить вход в систему

Главная последовательность:

1. В главном окне при нажатии кнопки «Список спектаклей и покупка билетов» открывается окно со списком запланированных спектаклей (рисунок 7)

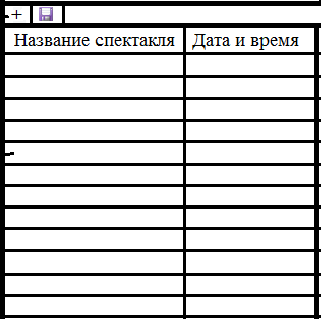


Рисунок 7 – Окно списка спектаклей

Название прецедента: Продажа билета зрителю

Действующее лицо: Кассир

Цель: Оформить билет для зрителя

Предусловие: Кассир должен выполнить вход в систему; должен существовать зритель с желанием купить билет; зритель должен быть зарегистрирован

Главная последовательность:

1. В главном окне при нажатии кнопки «Список спектаклей и покупка билетов» открывается окно со списком запланированных спектаклей.

2. При двойном нажатии левой кнопкой мыши по выбранной строке открывается окно с выбором типов билетов и количества мест (рисунок 8).

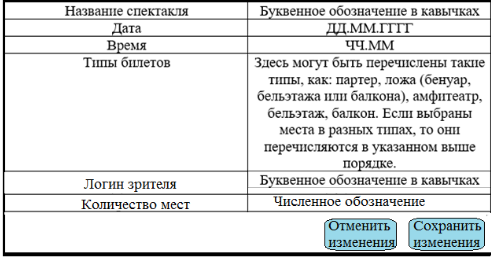


Рисунок 8 – Окно продажи билета

3. Вписывается логин пользователя в соответствующее поле. Заполняются остальные поля.

4. При нажатии кнопки «Сохранить изменения» данные о билете вносятся в файл данных о пользователях, а окно для покупки билета закрывается. Если нажимается кнопка «Отменить изменения», то данные о билете не вносятся в файл.

Название прецедента: Редактирование информации о спектаклях

Действующее лицо: Администратор

Цель: Редактировать данные о спектаклях

Предусловие: Администратор должен выполнить вход в систему

Главная последовательность:

1. В главном окне при нажатии кнопки «Список спектаклей и покупка билетов» открывается окно со списком запланированных спектаклей.

2. При двойном нажатии левой кнопкой мыши по выбранной ячейке с названием спектакля открывается окно с подробной информацией о спектакле (рисунок 9). Там же выполняется редактирование информации. Редактирование даты спектаклей производится без открытия данного окна, прямо в окне списка спектаклей.

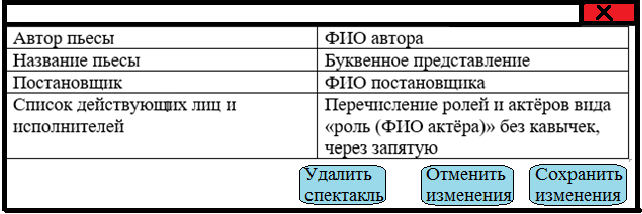


Рисунок 9 - Окно подробной информации о спектакле

3. Для добавления нового спектакля нужно нажать соответствующую кнопку в окне списка спектаклей.

Название прецедента: Просмотр списка актёров

Действующее лицо: Администратор или зритель

Цель: Просмотреть список актёров

Предусловие: Администратор или зритель должен выполнить вход в систему

Главная последовательность:

1. В главном окне при нажатии кнопки «Список актёров» откроется окно списка актёров (рисунок 10).



Рисунок 10 – Окно списка актёров

Название прецедента: Редактирование информации об актёрах

Действующее лицо: Администратор

Цель: Добавить информацию о новом актёре или изменить её у уже существующего

Предусловие: Администратор или должен выполнить вход в систему

Главная последовательность:

1. В главном окне при нажатии кнопки «Список актёров» откроется окно списка актёров.

2. Двойное нажатие левой кнопкой мыши по полю ФИО актёра откроет окно редактирования информации об актёре (рисунок 11).

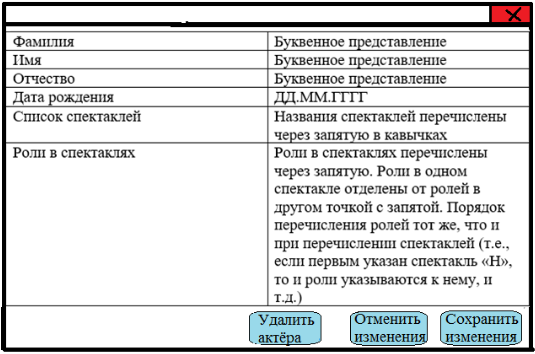


Рисунок 11 – Окно редактирования информации об актёре

3. После редактирования информации об актёре при нажатии кнопки «Сохранить изменения» данные об актёре будут перезаписаны. При нажатии кнопки «Отменить изменения» окно редактирования закроется, а данные не изменятся. При нажатии кнопки «Удалить актёра» все данные об актёре будут удалены.

# Список идентификаторов и их назначение

**3.1. Связи классов**

Для объединения идей о реализации программы в единую систему была спроектирована диаграмма классов (рисунок 12). На ней наглядно показаны связи между классами программы.

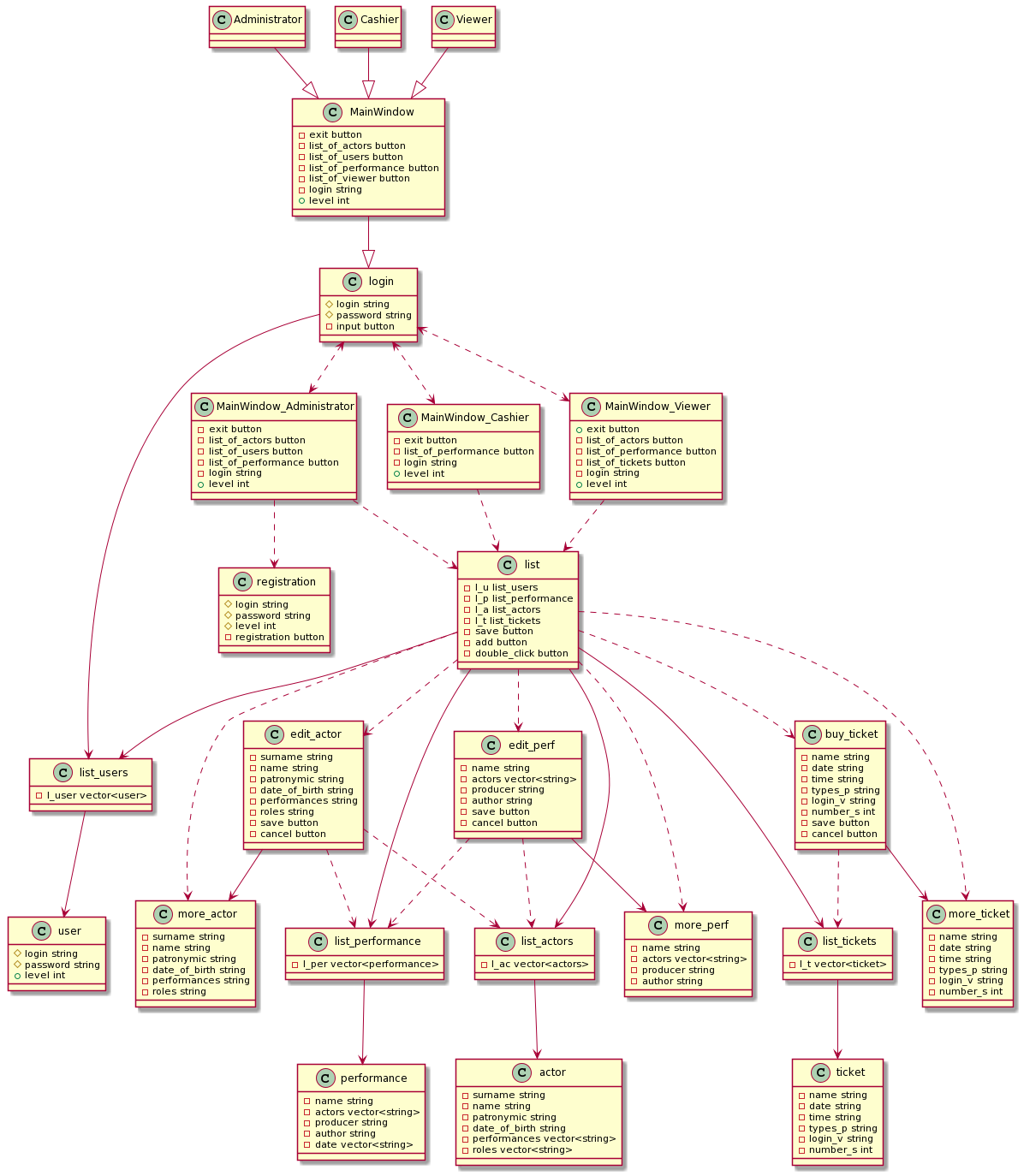


Рисунок 12 – Диаграмма классов

С виду диаграмма классов может показаться запутанной, но принцип работы итоговой программы прост. Т.к. в информационной системе необходимо отображать множество различной информации, было решено использовать так называемые «списки информации». Для хранения информации об актёрах, пользователях, выступлениях и билетах были созданы специальные классы: actor, user, performance и ticket соответственно. А т.к. каждый класс описывает лишь один объект (актёра, представление и т.д.), то для хранения всей информации были созданы вышестоящие классы, которые использовали вектора с упомянутыми выше классами. Получившиеся же классы-списки уже отображаются при открытии соответствующего окна программы. Также, зритель может узнать подробную информацию о том, или ином объекте (актёре, представлении, билете). Для предоставления этой информации используются те же окна, что и для редактирования (добавления) записи, но возможности изменения отключены

**3.2. Список файлов и их предназначение**

Файлы-классы представлений, актёров, пользователей и билетов соответственно:

* performance;
* actor;
* users;
* ticket.

Файлы-списки представлений, актёров, пользователей и билетов соответственно:

* l\_performance;
* list\_actor;
* list\_users;
* list\_tickets.

Файлы, в которых хранится информация о представлениях, актёрах, пользователях и билетах зрителей соответственно:

* performance.bin;
* actors.bin;
* users.bin;
* viewer.bin.

Окна добавления и подробной информации о билетах, актёрах и представлениях реализованы файлами:

* more\_ticket.bin;
* more\_actor.bin;
* more\_performances.bin.

Для входа в систему используется графическое окно, реализованное файлами login. Для регистрации нового пользователя используется графическое окно, реализованное файлами registration. Главное окно, через которое пользователи открывают окна с тем или иным списком, реализовано файлами mainwindow. Самая важная часть программы - окно, отображающее информацию из списка, реализовано файлами list.

# Подробное описание алгоритма

**4.1. Обоснование выбора алгоритма**

ИС «Театр» построена на принципе списков информации. В данных списках может выполняться поиск определённой записи, что позволяет их использовать как для отображения информации, так и для поиска определённой (например, логин, пароль и уровень доступа пользователя). В качестве контейнера для классов-представителей определённых объектов (пользователя, представления и т.д.) используются вектора из данных классов.

**4.2. Описание алгоритма**

Описание всех прецедентов представлено в пункте 2.2 данного документа. Здесь будет обобщённое представление о взаимодействии пользователя и ИС.

При запуске программы открывается окно входа в систему (login), а вместе с ним загружается список пользователей. Пользователь вбивает данные в соответствующие поля и, после поиска введённых данных в списке пользователей, входит в систему (если данные верные; в противном случае пользователь будет оставаться в окне входа в систему). Далее пользователь в зависимости от своего уровня доступа может выбрать в главном окне для открытия один из списков или выйти из системы. Так, для зрителя доступны все списки, кроме списка пользователей (список актёров, список представлений, список билетов вошедшего в систему пользователя). Для кассира доступен только список представлений для продажи билета зрителю. Для администратора доступны все списки, кроме списка билетов вошедшего пользователя (список актёров, список представлений, список пользователей). Администратор в данных списках может редактировать информацию – удалять пользователей актёров и представления, изменять информацию о представлениях и актёрах, добавлять новых пользователей, актёров или представления. Пользователь может лишь просматривать информацию в доступных ему списках (а также открывать подробную информацию). Кассир может только продавать билеты зарегистрированным пользователям-зрителям.

# Инструкция к программе

Инструкция к программе представлена в разделе 2.2 данного документа. В данном пункте подробно рассмотрены все возможные варианты взаимодействия с программой.

Заключение

Во время выполнения данной курсовой работы я успешно развил свои навыки объектно-ориентированного проектирования и программирования, а также улучшил знание среды разработки QtCreator. По итогу выполнения работы была создана успешно работающая программа информационной системы «Театр». Использование данной программы показало средний уровень успеха реализации – некоторые решения по проекту были незначительно изменены, а большая часть функционала успешно реализована.

Список использованных источников

1. СТО 4.2-07-2014 Система менеджмента качества. Общие требования к построению, изложению и оформлению документов учебной деятельности. – Введ. 30.12.2013. – Красноярск : ИПК СФУ, 2014. – 60 с. – Режим доступа: http://www.sfu-kras.ru/node/8127

2. Междисциплинарный курсовой проект базового уровня: учебно-методическое пособие [Текст] / сост. А.В. Редькина, А.В. Редькин – Красноярск: Сиб. федер. ун-т, 2016. – 40 с. – Режим доступа: <https://e.sfu-kras.ru/mod/resource/view.php?id=61924>

3. Документация к QtCreator [Электронный ресурс]: <https://doc.qt.io/qt-5/>