

Министерство науки и высшего образования Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«Национальный исследовательский университет ИТМО»

(Университет ИТМО)

Факультет программной инженерии и компьютерной техники

Образовательная программа Системное и прикладное программное обеспечение

Направление подготовки (специальность) Программная инженерия

О Т Ч Е Т

об производственной, технологической (проектно-технологической) практике

Тема задания: Разработка клиентского программного обеспечения

Обучающийся Патутин Владимир Михайлович, группа Р34101

Руководитель практики от профильной организации: Казанцев Максим Владимирович,
//TODO профессия, ООО «Риал Стэк»

Руководитель практики от университета: Маркина Татьяна Анатольевна, доцент факультета программной инженерии и компьютерной техники

Дата 13.04.2023

Санкт-Петербург

2023

Содержание

Содержание	2
1. Список терминов	3
2. Список сокращений и условных обозначений	4
3. Введение	5
3.1. Общая концепция	5
3.2. Решаемая проблема	5
4. Основная часть	6
4.1. Инструктаж обучающегося	6
4.2. Запуск проекта	6
4.3. Разбор архитектурных решений, изучение логики написанных компонентов, анализ code style	7
4.4. Повторяющийся этап ИЗ	8
Задача 1:	8
Метод создание таблицы в PunterAccessAuditPage	10
Метод обновления данных в таблице в PunterAccessAuditPage	10
Задача 2:	10
Метод создание таблицы в IPSearchControl	11
Метод обновления данных таблицы в IPSearchControl	12
Listener, определяющий необходимость вызова метода обновления данных	12
Задача 3:	12
Метод создание таблицы в CashOfficeListView	13
Создание listener-ов в CashOfficeListView	13
Вызовом метода addMenuItem в CashOfficeListView	14
5. Заключение	15
6. Список использованных источников	16
Приложение 1	17

1. Список терминов

Structured query language - декларативный язык программирования, применяемый для создания, модификации и управления данными в реляционной базе данных, управляемой соответствующей системой управления базами данных.

Java development kit - набор инструментов для разработки на языке Java. В него входят все компоненты, необходимые для того, чтобы программировать на этом языке.

Java - строго типизированный объектно-ориентированный язык программирования общего назначения.

Docker - это программная платформа для быстрой разработки, тестирования и развертывания приложений. Docker упаковывает ПО в стандартизованные блоки, которые называются контейнерами. Каждый контейнер включает все необходимое для работы приложения: библиотеки, системные инструменты, код и среду исполнения.

SWT - библиотека с открытым исходным кодом для разработки графических интерфейсов пользователя на языке Java.

Тим лидер - это «главный разработчик», который координирует деятельность остальных разработчиков и тех специалистов, которые также участвуют в разработке продукта, но не являются разработчиками.

Listener - это объект, уведомляемый о событии. Он должен быть зарегистрирован источником событий и реализовывать методы для получения и обработки уведомлений.

2. Список сокращений и условных обозначений

ИЗ - индивидуальное задание.

SQL - structured query language.

JDK - java development kit.

БД - база данных.

PM - product manager.

SWT - standard widget toolkit.

ТЗ - техническое задание.

3. Введение

3.1. Общая концепция

В рамках учебной практики была выбрана тема “Разработка клиентского программного обеспечения”. Для выполнения данной задачи будет проведен инструктаж работника, получение доступа к продукту для разработки, его запуск с последующим анализом code style, получение заданий для выполнения в проекте.

Целью этой темы является запуск проекта и выполнение ИЗ, которые связаны с улучшением или корректировкой работы данного проекта. Для достижения данной цели был выделен ряд задач:

- 1) Инструктаж обучающегося по ознакомлению с требованиями охраны труда, техники безопасности, пожарной безопасности, а также правилами внутреннего трудового распорядка.
- 2) Знакомство с коллективом, получение доступа к продукту для разработки, настройка окружения, запуск проекта.
- 3) Разбор архитектурных решений, изучение логики написанных компонентов, анализ code style.
- 4) Повторяющийся этап ИЗ:
 - 4.1) Получение задания.
 - 4.2) Работа над полученным заданием.
 - 4.3) Code review, в случае замечаний - рефакторинг.
 - 4.4) Тестирование, в случае обнаружения ошибок, несоответствующего поведения вернуться к пункту 2.
 - 4.5) Завершение задания.

3.2. Решаемая проблема

Улучшение или корректировка работы проекта поможет решить следующие проблемы: повышение удобства и привлекательности программы для пользователей, а также улучшение производительности, выявление уже существующих дефектов с их последующим

устранением, что в свою очередь увеличит доверие к системе, с последующим повышением конкурентоспособности.

4. Основная часть

4.1. Инструктаж обучающегося

В начале меня отвели в отдел кадров, где познакомили с работающими там людьми, попросили предоставить данные для заполнения документов и ознакомили с требованиями охраны труда, а также правилами внутреннего трудового распорядка. Потом меня отвели к специалисту по безопасности, который ознакомил с техникой безопасности и пожарной безопасностью. В конечном итоге, я ознакомился с инструктажем и подписал соответствующие документы.

4.2. Запуск проекта

На этом этапе мне был предоставлен доступ к проекту и инструкция описывающая шаги, необходимые для начала работы и настройки окружения разработчика с последующим запуском проекта. Для начала нужно было скачать необходимое ПО для работы с проектом:

1. Среда разработки приложения
2. Система контроля версий
3. Среда разработки на языке SQL, с возможностью администрирования баз данных
4. Инструмент для автоматической сборки проектов
5. Актуальный для проекта JDK
6. Приложение для разработки, доставки и запуска контейнерных приложений - docker
7. Групповой мессенджер и свободный текстовый редактор

Следующим шагом стала настройка окружения, включающая в себя установку системных переменных, включение WSL2, установку всего вышеперечисленного ПО, а также дополнительных программ, которые облегчают процесс разработки именно мне:

1. Far Manager

2. ConEmu
3. IntelliJ IDEA
4. Skype
5. Terminal

Теперь можно приступить к запуску клиентской части приложения, для этого необходимо скачать проект с удаленного репозитория и настроить среду разработки для соответствия окружения кодировкам, указания пути к jdk и инструменту сборки проекта, установка файла code style.

Следующим шагом стало импортирование проекта в среду разработки, настройка его отдельных модулей и сборка всех проектов, чтобы обновить jar файлы и убедиться в отсутствии ошибок компиляции. После необходимо установить конфигурацию для автоматического запуска клиентского приложения.

И наконец, можно приступить к поднятию серверной части приложения, для этого необходимо скачать еще один проект, который представляет из себя набор скриптов, помогающих генерировать файлы, создавать локальные сборки, запускать docker-контейнеры буквально одним кликом и имеет хороший конспект, с которым я ознакомился в свободное время, по командам docker. Читаем README.md файл и по инструкции настраиваем docker, после перезапускаем компьютер для применения настроек и выполняем скрипты для создания контейнеров и инициализации локальной БД. Вызываем скрипт для отображения запущенных docker-контейнеров и можем убедиться, что все они работают на данный момент.

Таким образом мной был получен доступ к проекту, настроено окружение и запущен проект, о чем было рассказано на еженедельном встрече с РМ, где я и познакомился с командой разработки.

4.3. Разбор архитектурных решений, изучение логики написанных компонентов, анализ code style

Данный этап подразумевает прочтение workflow и изучение кода проекта, над которым я буду работать.

Прочтение workflow достаточно сильно помогло разобраться в актуальных вопросах относительно проекта, таких как:

1. Базовое знакомство с Confluence
2. Базовое знакомство с Jira
3. Базовое знакомство с BitBucket
4. Базовое знакомство с Graylog
5. Действующие ветки, в которых ведется разработка, тестирование, продакшн ветки, правильное наименование веток, решение конфликтов при слиянии веток.
6. Паттерн написания коммитов
7. Создание задач, создание подзадач.

Также, благодаря workflow я смог понять архитектурные решения проекта. В данный момент проект является монолитом, который постепенно переносится на SOA архитектуру, для более удобной работы с ним.

При изучении логики написанных компонентов, я смог подметить для себя много новой информации, а именно увидеть реализацию разных шаблонов проектирования, использования многопоточного и асинхронного выполнения потоков, посмотреть на правильное высвобождение ресурсов по окончании выполнения определенного блока кода. Помимо этого, у меня была прекрасная возможность познакомиться с основной библиотекой библиотекой проекта - SWT.

4.4 Повторяющийся этап ИЗ

Теперь подняв проект и примерно понимая его структуру, стало возможна разработка клиентского программного обеспечения. Согласно уставу, если не предусмотрено текущих задач, необходимо связаться с тим-лидером и запросить назначение новой задачи, к которой впоследствии можно приступить.

Задача 1:

ТЗ: Заменить класс ArrayTableViewer на NatTableViewer в перспективе “Игроки” во вкладке “Доступ”.

Решение данной задачи я начал с анализа класса ArrayTableViewer и методов, которые используются в данной перспективе. ArrayTableViewer класс является старой реализацией представления таблицы, используемой в проекте и строится на использовании библиотеки

SWT. Сам класс имеет огромную функциональность, поэтому я опишу три главных метода на мой взгляд - конструктор класса, метод `setTableData`. Конструктор данного класса имеет несколько аргументов, а именно:

1. Входные данные, которые должны отображаться в таблице.
2. Имена колонок таблицы - поступают в виде массива.
3. Типы колонок таблицы, которым должны соответствовать входные данные - поступают в виде массива.
4. Значение отвечающие за возможность редактирования колонок таблицы - поступают в виде массива.

Метод `setTableData` имеет один аргумент - список, который содержит входные данные, они будут отображаться в таблице. Если какие-то данные ранее содержались в таблице, то они будут перезаписаны, без возможности восстановления.

Следующим шагом стало изучение класса `NatTableViewer`. Сам класс представляет более удобную реализацию класса `Table` из библиотеки SWT, благодаря заранее реализованной сортировке строк в таблице по столбцам, а также встроенной выгрузке полей таблицы в pdf, xls, csv, xlsx. Также функционал данного класса достаточно обширный и поиск функций аналогичных функциям `ArrayTableViewer` не составляет труда. Метод аналогичный `setTableData` из `ArrayTableViewer` называется `setTableData` и имеет аналогичные аргументы и очень похожую логику.

С попытками найти конструктор, имеющий аналогичные аргументы, возникли проблемы. Решением стало обнаружение ранее написанного кода, который связан с созданием объекта `NatTableViewer`. Благодаря этому коду, можно понять, что необходимо использовать конструктор `NatTableViewer`, который принимает в себя в качестве аргумента элемент класса `NatColumnsDefinition`.

Таким образом, можно начать поиск классов, которые необходимо заменить, и приступить к замене класса `ArrayTableViewer` на `NatTableViewer`. В данной перспективе нужно было заменить только 1 класс - `AccessAuditPage`.

Метод создание таблицы в PunterAccessAuditPage

Метод обновления данных в таблице в *PunterAccessAuditPage*

Задача 2:

Данная задача достаточно сильно похожа на сделанную ранее, но имеет существенное отличие, в данной реализации `ArrayTableView` связан с `UserAlertsIPSearchPreferencePage` классом, который позволяет проводить манипуляции с колонками, а именно переставлять их местами, удалять, добавлять, если колонка ранее была удалена. Также нам нужно разобраться в работе метода `getRowData`.

Метод `getRowData` имеет один аргумент - номер строки, информацию о которой нужно вернуть, а также использует библиотеку SWT для получения нужной информации о выбранном объекте. Для лучшего понимания логики данного метода, я обратился к документации библиотеки SWT и изучил некоторые метод из класса `Table`[1].

Найти аналог функции `getRowData` не составило труда, данный метод называется `getSelection` и не имеет аргументов. Благодаря отсутствию аргументов, он имеет облегченную логику вызова, потому что ранее необходимо было сначала получить индекс выбранного элемента и только после появлялась возможность получить данные этого элемента. Теперь же нам достаточно просто вызвать метод `getSelection`.

При поиска возможности связывания NatTableView и UserAlertsIPSearchPreferencePage, я нашел класс NatPreferencesTable. Он принимает в себя объект NatTableView и позволяет использовать весь функционал, который ранее предоставлялся класс UserAlertsIPSearchPreferencePage, без указания на него. Таким образом, появляется возможность полного удаления класса UserAlertsIPSearchPreferencePage, с незначительной корректировкой класса, который ранее использовал ArrayTableView.

Теперь можно начать поиск классов, которые необходимо заменить, и приступить к их замене. В данной перспективе нужно было заменить только 1 класс - IPSearchControl.

[illegible]

Метод создание таблицы в IPSearchControl

Начнем решение данной задачи с NatTableView, нам нужно обратить внимание на метод addItem, который имеет два аргумента:

1. Название, которое будет отображаться в элементе меню.
2. Listener, который будет вызван при нажатии.

Таким образом, решение данной задачи сводится к переводу NSArrayTableView на NatTableView, с последующей реализацией Listener-ов, логика которых будет заключаться в открытии отдельного окна с данными. И вызовом метода addItem, который примет название и реализацию listener-a. Приступим к решению:

```

private void createTableViewController()
{
    NSArrayTableView tableView = new NSArrayTableView();
    tableView.dataSource = new NSArrayDataSource();
    tableView.delegate = new NSArrayDelegate();
    tableView.backgroundColor = UIColor.White;
    tableView.separatorStyle = UITableViewCellSeparatorStyle.None;
    tableView.registerClass(UITableViewCell.class, forCellReuseIdentifier: "cell");

    NSArray<UITableViewCell> cells = new NSArray<UITableViewCell>();
    UITableViewCell cell1 = new UITableViewCell();
    cell1.textLabel.text = "1";
    cells.addObject(cell1);

    UITableViewCell cell2 = new UITableViewCell();
    cell2.textLabel.text = "2";
    cells.addObject(cell2);

    UITableViewCell cell3 = new UITableViewCell();
    cell3.textLabel.text = "3";
    cells.addObject(cell3);

    tableView.reloadData();
    tableView.delegate = new NSArrayDelegate();
    tableView.dataSource = new NSArrayDataSource();
}

```

Метод создание таблицы в CashOfficeListView

```

private void createTableViewController()
{
    NSArrayTableView tableView = new NSArrayTableView();
    tableView.dataSource = new NSArrayDataSource();
    tableView.delegate = new NSArrayDelegate();
    tableView.backgroundColor = UIColor.White;
    tableView.separatorStyle = UITableViewCellSeparatorStyle.None;
    tableView.registerClass(UITableViewCell.class, forCellReuseIdentifier: "cell");

    NSArray<UITableViewCell> cells = new NSArray<UITableViewCell>();
    UITableViewCell cell1 = new UITableViewCell();
    cell1.textLabel.text = "1";
    cells.addObject(cell1);

    UITableViewCell cell2 = new UITableViewCell();
    cell2.textLabel.text = "2";
    cells.addObject(cell2);

    UITableViewCell cell3 = new UITableViewCell();
    cell3.textLabel.text = "3";
    cells.addObject(cell3);

    tableView.reloadData();
    tableView.delegate = new NSArrayDelegate();
    tableView.dataSource = new NSArrayDataSource();
}

```

Создание listener-ов в CashOfficeListView



Вызовом метода `addMenuItem` в `CashOfficeListView`

Рефакторинга написанный код не потребовал. Тестирование прошло успешно. Задача закрыта.

5. Заключение




В период производственной практики мне удалось не только выполнить поставленные задачи, благодаря которым улучшилась визуальная составляющая и удобство интерфейса клиентской программы, но и попробовать себя в корпоративной разработке, ознакомившись с процессом разработки и сопровождения программного обеспечения, который включает в себя различные этапы, такие как планирование, анализ, проектирование, разработка, тестирование и внедрение. А также я смог успешно применить мои ранее полученные знания из университета в реальном проекте, что является важной частью моего профессионального роста.

6. Список использованных источников




1. Table (Eclipse Platform API Specification). Режим доступа:
<https://archive.eclipse.org/eclipse/downloads/documentation/2.0/html/plugins/org.eclipse.platform.doc.isv/reference/api/org/eclipse/swt/widgets/Table.html>, свободный.

Приложение 1

Вид интерфейса и его функционала до внесения изменений в классе
CashOfficeListView:

#	Name	
1	test	<div> New cash office</div> <div> Edit cash office</div> <div> Delete cash office</div>

Вид интерфейса и его функционала после внесения изменений в классе
CashOfficeListView:

#	Name	
1	test	<div> New cash office</div> <div> Edit cash office</div> <div> Delete cash office</div> <div>Restore table columns</div> <div>Export to Excel (XLS)</div> <div>Export to Excel (XLSX)</div> <div>Export to Excel (CSV)</div> <div>Export to PDF (PDF)</div> <div>Export selected to PDF (PDF)</div> <div>Copy to clipboard</div> <div>Drop clipboard selection</div>