**Министерство образования и науки**
**Российской Федерации.**

**Московский авиационный институт**
**(национальный исследовательский университет)**

# Журнал

# ПО ПРОИЗВОДСТВЕННОЙ ПРАКТИКЕ

Наименование практики:*вычислительная/исследовательская*
Студент:Лагода
Д.А.И.
Факультет№8,курс 2,
группа 7
**Практика с 29.06.20 по 12.07.20**

**Москва,2020**

## Оглавление



**ЗАДАНИЕ**
**кафедры 806 по вычислительной/исследовательской практике:**

Десктопное приложение на Java, симулятор биржи, сохранение в PostgreSQL.

## Теоретическое объяснение техники.

Java — объектно-ориентированный язык программирования, разрабатываемый компанией Sun Microsystems с 1991 года и официально выпущенный 23 мая 1995 года. Изначально новый язык программирования назывался Oak (James Gosling) и разрабатывался для бытовой электроники, но впоследствии был переименован в Java и стал использоваться для написания апплетов, приложений и серверного программного обеспечения.
Программы на Java могут быть транслированы в байт-код, выполняемый на виртуальной java-машине (JVM) — программе, обрабатывающей байт-код и передающей инструкции оборудованию, как интерпретатор, но с тем отличием, что байт-код, в отличие от текста, обрабатывается значительно быстрее.

Язык Java зародился как часть проекта создания передового программного обеспечения для различных бытовых приборов. Реализация проекта была начата на языке C++, но вскоре возник ряд проблем, наилучшим средством борьбы с которыми было изменение самого инструмента — языка программирования. Стало очевидным, что необходим платформо-независимый язык программирования, позволяющий создавать программы, которые не приходилось бы компилировать отдельно для каждой архитектуры и можно было бы использовать на различных процессорах под различными операционными системами.

Язык Java потребовался для создания интерактивных продуктов для сети Internet. Фактически, большинство архитектурных решений, принятых при создании Java, было продиктовано желанием предоставить синтаксис, сходный с C и C++. В Java используются практически идентичные соглашения для объявления переменных,

передачи параметров, операторов и для управления потоком выполнением кода. В Java добавлены все хорошие черты C++.

Три ключевых элемента объединились в технологии языка Java

•Java предоставляет для широкого использования свои апплеты (applets) — небольшие, надежные, динамичные, не зависящие от платформы активные сетевые приложения, встраиваемые в страницы Web. Апплеты Java могут настраиваться и распространяться потребителям с такой же легкостью, как любые документы HTML

•Java высвобождает мощь объектно-ориентированной разработки приложений, сочетая простой и знакомый синтаксис с надежной и удобной в работе средой разработки. Это позволяет широкому кругу программистов быстро создавать новые программы и новые апплеты

•Java предоставляет программисту богатый набор классов объектов для ясного абстрагирования многих системных функций, используемых при работе с окнами, сетью и для ввода-вывода. Ключевая черта этих классов заключается в том, что они обеспечивают создание независимых от используемой платформы абстракций для широкого спектра системных интерфейсов

Внешние программы формируют запросы к СУБД на языке управления данными Structured Query Language. Что такое SQL и чем отличается от привычных языков программирования? Одна из особенностей SQL – декларативность. То есть, **SQL — декларативный язык**. Это значит, что, вбивая команды, то есть, создавая запросы к SQL-серверу, мы описываем, что именно хотим получить, а не каким способом. Посылая серверу запрос SELECT * FROM CUSTOMER (приблизительный перевод с SQL на русский: *«сделать выборку из таблицы COSTUMER, выборка состоит из всех строк таблица»*), мы получим данные по всем пользователям. Совершенно неважно, как и откуда сервер загрузит и сформирует интересующие нас данные. Главное – правильно сформулировать запрос.

•Что такое SQL-Сервер и как он работает? Взаимодействие с СУБД происходит по клиент-серверному принципу. Некая внешняя программа посылает запрос в виде операторов и команд на языке SQL, СУБД его обрабатывает и высылает ответ. Для упрощения примем, что SQL Сервер = СУБД.

JDBC

В 80-е годы прошлого века персональные компьютеры типа PC XT/AT завоевали рынок. Во многом это произошло благодаря модульности их конструкции. Это означает, что пользователь мог довольно просто менять ту или иную составную

часть своего компьютера (процессор, видеокарту, диски и тому подобное). Это замечательное свойство сохранилось и поныне: мы меняем видеокарту и обновляем драйвер (иногда он и вовсе обновляется сам, в автоматическом режиме). Чаще всего при таких манипуляциях ничего плохого не происходит, и существующие программы продолжат работать с обновившейся системой без переустановки. Аналогично и для работы в Java с СУБД. Для стандартизации работы с SQL-серверами взаимодействие с ней можно выполнять через единую точку — **JDBC** (Java DataBase Connectivity). Она представляет собой реализацию пакета *java.sql* для работы с СУБД. Производители всех популярных SQL-серверов выпускают для них драйверы JDBC. Приложение использует экземпляры классов из *java.sql*. Затем мы передаем необходимые команды для получения/модификации данных. Далее *java.sql* через *jdbc-драйвер* взаимодействует с СУБД и возвращает нам готовый результат.

Архитектура

Само приложение состоит из двух связанных частей, базы данных PostgreSQL, созданной  и определенной заранее и Java- частью реализующей работу с базами данных при помохи соответсвующего драйвера.

SQL часть состоит из работы с SQL коммандами типа «CREATE TABLE OperLogTable( id integer CONSTRAINT Operid PRIMARY KEY,Stock varchar(10), type varchar(8), operTime varchar(8),money double precision, shareOverall integer);» и «insert into ShareTable (id,nameShare,PercCourseShare,startPrice,MaxPercent) values (4,'Sony',9,10.0,45);» для создания и заполнения таблиц. Мы имеем две таблицы под названиями OperLogTable и ShareLog. Таблица ShareLog является нашей «настроечной» таблицей, из которой наша программа берет первоначальные данные для запуска программы. OperLogTable по сути является пустой таблицей, которую мы заполняем по мере проведения работы, записывая туда лог проведенных операций и получая оттуда данные для динамической таблице, соответствующей ей.

Java часть гораздо сложней и состоит из драйвера, который мы скачали заранее для подключения и работе с БД и классов. Главным классом является StockMarket? который «чистит» записи в OperLogTable, оставшиеся с прошлого раза и вызывет основное «тело» приложения : TextInputFrame. Данный класс включает в себя все необходимые нам методы и процедуры. Основной метод данного класса вызывает окно, в котором находятся необходимые элементы меню: кнопки купить/продать, таблица акций, таблица денег и лог операций.

Кнопки считывают нажатие на них и открывают доп_окно в котором следует задать необходимое количество акций и подтвердить выбор. Выбор акций для покупки осуществляется путем нажатия на необходимую строку таблицы

акций.После проверки количества денег или акций, производится изменение количества денег и акций и записывается данные об операции путем работы с OperLogTable SQL таблицей.

Таблица акций начинает работать от TableInit(Vector<Vector<String>> data,Vector<String> header) в котором мы считываем данные записанные в изначальной таблице SQL, после этого мы ставим метод RandomStock, посредством которого мы высчитываем курс и новые цены для таблицы. Кроме того получая новые данные о цене акций каждой компании, мы изменяем соответствующие данные в «денежной» таблице. Обновление таблиц ( и их рандомайзер) происходит в периодичность 10 секунд, посредством таймера.

Таблица денег представляет собой таблицу 1x5, в который первая ячейка показывает имеющиеся деньги в кэше, а другие ячейки показывает цену акций компаний соответствующим им, которые принадлежат нам.

Лог операций можно назвать гридом, т. к. мы создали его путем написания модели таблицы, при помощи DefaultTableModel, и он полностью связан с соответсвующей ему таблицей в SQL. Сама таблица получает данные из SQL таблицы, после того как таблица в SQL была изменена, т.е. делая таблицу неограниченной, путем добавления ползунка прокрутки путем JScrollPane.

Помимо данных основных элементов и их общего описания в процессе работы программы происходят другие, более незаметные и многоочисленные процессы внутри самих этих методов, в том числе работающих с БД.

Рационализация.

По последующим направлениям улучшения данной программы я разработал ряд предложений:

1. Обеспечение возможности  настраивать ShareTable непосредственно из приложения.

2.Создание соответсвующих моделей DefaultTableModel для CurrentMoney и Share таблиц, т.е. их синхронизация в количестве их ячеек и возможность создавать таблица неограниченной длины и разнообразия.

3.Подстветка строк в ShareTable красным или зеленым цветом, тех компаний, чей курс соответственно падает или растет.

4.Введение возможности поставить приложение на паузу.

Вывод.

В процессе работы над данной задачей я научился многому. Мое знание ООП программирования в Си языках нашло здесь место для развития и углубления. Я научился работать с GUI интерфейсами и создавать их, познакомился с SQL языком,

его командами и базами данных, научился работать с базами данных посредством сторонних средств, в данном случае при помощи языка Java.