ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ (НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)»

Журнал практики

Студента __Андрианова Елена Андреевна_____
(фамилия, имя, отчество)

Институт №3 «Системы управления, информатика и электроэнергетика»___

Кафедра № 319 «Системы интеллектуального мониторинга»_

Учебная группа__М3О-135Б-20____

Направление подготовки 09.03.01
(шифр)
«Информатика и вычислительная техника»
(название направления)

Вид практики _____*учебная*_____
(учебной, производственной, преддипломной или другой вид практики)

Руководитель практики от МАИ

Зеленова Марина Викторовна _____
(фамилия, имя, отчество) *(подпись)*

_____/_____/ "18" июня 2021 г.
(подпись студента) *(дата)*

Москва 2021

1.Место и сроки проведения практики

Сроки проведения практики:

*-дата начала практики*_____*29 июня 2021 г.*_____

*-дата окончания практики*___*12 июля 2021 г.*_____

Наименование предприятия *кафедра 319 «Системы интеллектуального мониторинга»*

Название структурного подразделения (отдел, лаборатория)

_____ _____

2. Инструктаж по технике безопасности

1. Вводный инструктаж.

Проводится со следующими категориями слушателей:

- все сотрудники, принимаемые на работу;
- работники, прибывшие в организацию в командировку, и работники других организаций, которые заняты на определенных участках;
- для обучающихся в учебных заведениях, которые направлены в организацию на производственную практику;
- другие лица, занятые в производственной сфере работы предприятия.

Включает:

- инструктаж по технике безопасности на рабочем месте;
- инструктаж по охране труда;
- описание направления деятельности предприятия и отдела.

2. Первичный инструктаж на рабочем месте.

Должен проводиться перед тем, как сотрудник начнет самостоятельную работу. Его обязаны прослушать:

- все принятые на предприятие сотрудники, сюда относятся и те, кто заключил с организацией трудовой договор, работают на дому, а также по совместительству;
- сотрудники, переведенные с одного рабочего места на другое либо выполняющим данный вид работ впервые;

- командированные из других предприятий, временные работники, учащиеся образовательных учреждений, направленные на производственную практику, и другие сотрудники, чья работа связана с производственной деятельностью организации.

Включает:

- инсталляцию программного и аппаратного обеспечения для информационных и автоматизированных систем для анализа данных и машинного обучения;
- проверку состояния вычислительного оборудования;
- осуществление необходимых профилактических процедур вычислительного оборудования.

Шевченко Д.А. _____/ “29”июня 2021 г.
(подпись проводившего) *(дата проведения)*

3. Индивидуальное задание студенту

Предварительное определение темы и объема работ

1. Изучение систем контроля версий (Git, SVN и др.), принципов работы с ними. Создание репозитория на https://github.com;

2. Изучение выбранных технологий (см. список тем) и реализация примеров использования выбранных технологий, а именно: Инструменты разработки приложений с оконным интерфейсом на java.swing. Реализованные примеры должны быть размещены в своем созданном репозитории на https://github.com.

“29” июня 2021 г.

(дата проведения)

4.План выполнения индивидуального задания

План работ

1) Изучение материала по теме задания.

2) Создание аккаунта на https://github.com/ .

3) Определение конечной цели.

4) Создание конечной версии программы.

5) Тестирование конечной версии.

6) Выгрузка программы на https://github.com/ .

Руководитель практики от МАИ: _Зеленова М.В.___/_____/

_____/_____/ “_29_” _июня__ 2021 г.
(подпись студента) *(дата)*

6.Отзыв руководителя практики от МАИ

За время прохождения практики Андрианова Е.А. зарекомендовал себя как ответственный и исполнительный практикант, показал высокий уровень теоретической подготовки, хорошее умение применить и использовать полученные знания для решения поставленных перед ним задач.

Программа практики выполнена полностью.

В целом работа практиканта Андрианова Е.А. заслуживает оценки «_____».

Материалы, изложенные в отчете студента, полностью соответствуют индивидуальному заданию.

За время прохождения практики были сформированы следующие компетенции:

Шифр	**Компетенция**
ПК-1	Способность разрабатывать модели компонентов информационных систем, включая модели баз данных
ПК-3	Способность обосновывать реализуемые проектные решения, осуществлять постановку и выполнение экспериментов по проверке корректности и эффективности эт
ПК-7	Способность проверять техническое состояние вычислительного оборудования и осуществлять необходимые профилактические процедуры

И соответствующие им результаты освоения.

Руководитель практики от МАИ: _Зеленова М.В.___/_____/

(фамилия, имя, отчество) *подпись)*

«_12_»_июля_2021 г

.

6. Отчет студента о практике

В процессе прохождения практики мною были выполнены следующие задания:

1) Составление технического задания.
2) Изучение наиболее популярных методов и алгоритмов оконного интерфейса swing.
3) Изучение возможностей оконного приложения, написанного на библиотеке swing;
4) Написание игры с использованием библиотеки swing.
5) Тестирование полученного приложения.
6) Создание репозитория на https://github.com/ с проектом.

Понятие оконного интерфейса

Оконный интерфейс – способ организации полноэкранного интерфейса программы, в котором каждая интегральная часть располагается в окне – собственном субэкранном пространстве, находящемся в произвольном месте «над» основным экраном. Несколько окон, одновременно располагающихся на экране, могут перекрываться, виртуально находясь «выше» или «ниже» друг относительно друга.

Оконный интерфейс реализуется как в графическом, так и в текстовом режиме. Наиболее известной (неполной) реализацией оконного интерфейса в текстовом режиме является программа-оболочка Питера Нортона «Norton Commander» и её многочисленные модификации.

Библиотека awt

Первой попыткой Sun создать графический интерфейс для Java была библиотека AWT (Abstract Window Toolkit) — инструментарий для работы с различными оконными средами. Sun сделал прослойку на Java, которая вызывает методы из библиотек, написанных на языке C. Библиотечные методы AWT создают и используют графические компоненты операционной среды. С одной стороны, это хорошо, так как программа на Java похожа на остальные программы в рамках одной ОС. Но при запуске ее на другой платформе могут возникнуть различия в размерах компонентов и шрифтов, которые будут портить внешний вид программы.

Чтобы обеспечить мультиплатформенность AWT, интерфейсы вызовов компонентов были унифицированы, вследствие чего их функциональность получилась немного урезанной. Да и набор компонентов получился довольно небольшой. Так, например, в AWT нет таблиц, а в кнопках не поддерживается отображение иконок. Тем не менее пакет java.awt входит в Java с самого первого выпуска и его можно использовать для создания графических интерфейсов.

Таким образом, компоненты AWT не выполняют никакой "работы". Это просто «Java-оболочка» для элементов управления той операционной системы, на которой они работают. Все запросы к этим компонентам перенаправляются к операционной системе, которая и выполняет всю работу.

Использованные ресурсы AWT старается освобождать автоматически. Это немного усложняет архитектуру и влияет на производительность. Написать что-то серьезное с использованием AWT затруднительно. Сейчас ее используют разве что для апплетов.

Abstract Window Toolkit (AWT) – исходная платформо-независимая оконная библиотека графического интерфейса (Widget toolkit). Некоторые разработчики предпочитают эту модель, поскольку она обеспечивает высокую степень соответствия

основному оконному инструментарию и беспрепятственную интеграцию с родными приложениями. Другими словами, GUI программа, написанная с использованием AWT, выглядит как родное приложение Microsoft Windows, будучи запущенной на Windows, и в то же время как родное приложение Apple Macintosh, будучи запущенным на Mac, и т. д.. Однако, некоторым разработчикам не нравится эта модель, потому что они предпочитают, чтобы их приложения выглядели одинаково на всех платформах.

Библиотека swing

Вслед за AWT Sun разработала графическую библиотеку компонентов Swing, полностью написанную на Java. Для отрисовки используется 2D, что принесло с собой сразу несколько преимуществ. Набор стандартных компонентов значительно превосходит AWT по разнообразию и функциональности. Swing позволяет легко создавать новые компоненты, наследуясь от существующих, и поддерживает различные стили и скины.

Создатели новой библиотеки пользовательского интерфейса Swing не стали «изобретать велосипед» и в качестве основы для своей библиотеки выбрали AWT. Конечно, речь не шла об использовании конкретных тяжеловесных компонентов AWT (представленных классами Button, Label и им подобными). Нужную степень гибкости и управляемости обеспечивали только легковесные компоненты. На рисунке 1 представлена связь между AWT и Swing.

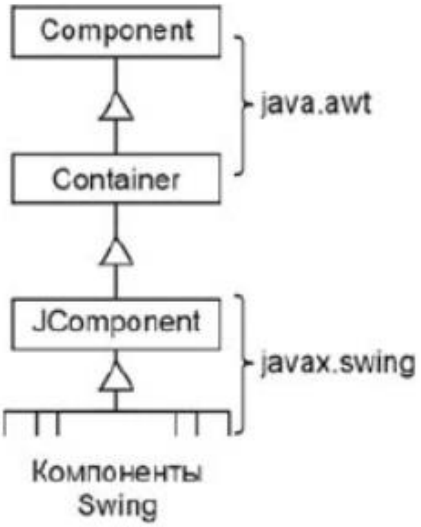

Рис. 1 – диаграмма наследования связи AWT и Swing

Swing – библиотека для создания графического интерфейса. Разработана компанией Sun Microsystems. Содержит ряд графических компонентов (англ. Swing widgets), таких как кнопки, поля ввода, таблицы и т. д. Архитектура Swing разработана таким образом, что вы можете изменять «look and feel» (L&F) вашего приложения. «Look» определяет внешний вид компонентов, а «Feel» – их поведение.

Swing была представлена миру в 1997 году, как технология, призванная решить проблемы AWT. Компоненты Swing часто определяют термином lightweight.

Swing контейнеры высшего уровня

Для создания графического интерфейса приложения необходимо использовать специальные компоненты библиотеки Swing, называемые контейнерами высшего уровня (top level containers). Они представляют собой окна операционной системы, в которых размещаются компоненты пользовательского интерфейса. К контейнерам высшего уровня относятся окна JFrame и JWindow, диалоговое окно JDialog, а также апплет JApplet (который не является окном, но тоже предназначен для вывода интерфейса в браузере, запускающем этот апплет). Контейнеры высшего уровня Swing представляют

собой тяжеловесные компоненты и являются исключением из общего правила. Все остальные компоненты Swing являются легковесными.

Конструктор JFrame() без параметров создает пустое окно. Конструктор JFrame(String title) создает пустое окно с заголовком title. Чтобы создать простейшую программу с пустым окном необходимо использовать следующие методы:

- setSize(int width, int height) - определение размеров окна;
- setDefaultCloseOperation(int operation) - определение действия при завершении программы;
- setVisible(boolean visible) - сделать окно видимым.

Если не определить размеры окна, то оно будет иметь нулевую высоту независимо от того, что в нем находится. Размеры окна включают не только «рабочую» область, но и границы и строку заголовка.

Метод setDefaultCloseOperation определяет действие, которое необходимо выполнить при "выходе из программы". Для этого следует в качестве параметра operation передать константу EXIT_ON_CLOSE, описанную в классе JFrame.

По умолчанию окно создается невидимым. Чтобы отобразить окно на экране вызывается метод setVisible с параметром true. Если вызвать его с параметром false, окно станет невидимым.

Для подключения библиотеки Swing в приложении необходимо импортировать библиотеку javax.swing.

Корневая панель JRootPane

Каждый раз, как только создается контейнер высшего уровня, будь то обычное окно, диалоговое окно или апплет, в конструкторе этого контейнера создается корневая панель JRootPane. Контейнеры высшего уровня Swing следят за тем, чтобы другие компоненты не смогли "пробраться" за пределы JRootPane.

Корневая палель JRootPane добавляет в контейнеры свойство "глубины", обеспечивая возможность не только размещать компоненты один над другим, но и при необходимости менять их местами, увеличивать или уменьшать глубину расположения компонентов. Такая возможность необходима при создании многодокументного приложения Swing, у которого окна представляют легковесные компоненты, располагающиеся друг над другом, а также выпадающими (контекстными) меню и всплывающими подсказками.

На рисунке 2 наглядно представлена структура корневой панели JRootPane.

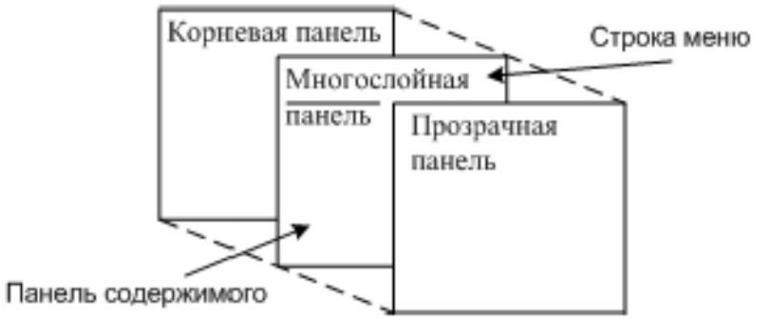

Рис. 2 – структура корневой папки

Корневая панель JRootPane представляет собой контейнер, унаследованный от базового класса Swing JComponent. В этом контейнере за расположение компонентов отвечает специальный менеджер расположения, реализованный во внутреннем классе RootPaneLayout. Этот менеджер расположения отвечает за то, чтобы все составные части корневой панели размещались так, как им следует: многослойная панель занимает все

пространство окна; в ее слое FRAME_CONTENT_LAYER располагаются строка меню и панель содержимого, а над всем этим располагется прозрачная панель.

Все составляющие корневой панели JRootPane можно получить или изменить. Для этого у нее есть набор методов get/set. Программным способом JRootPane можно получить с использованием метода getRootPane().

Кроме контейнеров высшего уровня корневая панель применяется во внутренних окнах JInternalFrame, создаваемых в многодокументных приложениях и располагающихся на "рабочем столе" JDesktopPane. Это позволяет забыть про то, что данные окна представляют собой обычные легковесные компоненты, и работать с ними как с настоящими контейнерами высшего уровня.

Панель содержимого ContentPane

Панель содержимого ContentPane – это следующая часть корневой панели, которая используется для размещения компонентов пользовательского интерфейса программы. ContentPane занимает большую часть пространства многослойной панели (за исключением места, занимаемого строкой меню). Чтобы панель содержимого не закрывала добавляемые впоследствии в окно компоненты, многослойная панель размещает ее в специальном очень низком слое с названием FRAME_CONTENT_LAYER, с номером -30000.

Обратиться к панели содержимого можно методом getContentPane() класса JFrame. С помощью метода add(Component component) можно добавить на нее любой элемент управления. Заменить ContentPane любой другой панелью типа JPanel можно методом setContentPane().

Многослойная панель JLayeredPane

В основании корневой панели (контейнера) лежит так называемая многослойная панель JLayeredPane, занимающая все доступное пространство контейнера. Именно в этой панели располагаются все остальные части корневой панели, в том числе и все компоненты пользовательского интерфейса.

JLayeredPane используется для добавления в контейнер свойства глубины (depth). То есть, многослойная панель позволяет организовать в контейнере третье измерение, вдоль которого располагаются слои (layers) компонента. В обычном контейнере расположение компонента определяется прямоугольником, который показывает, какую часть контейнера занимает компонент. При добавлении компонента в многослойную панель необходимо указать не только прямоугольник, занимаемый компонентом, но и слой, в котором он будет располагаться. Слой в многослойной панели определяется целым числом. Чем больше определяющее слой число, тем выше слой находится.

Первый добавленный в контейнер компонент оказывается выше компонентов, добавленных позже. Чаще всего разработчик не имеет дело с позициями компонентов. При добавлении компонентов их положение меняются автоматически. Тем не менее многослойная панель позволяет менять позиции компонентов динамически, уже после их добавления в контейнер.

Возможности многослойной панели широко используются некоторыми компонентами Swing. Особенно они важны для многодокументных приложений, всплывающих подсказок и меню. Многодокументные Swing приложения задействуют специальный контейнер JDesktopPane («рабочий стол»), унаследованный от JLayeredPane, в котором располагаются внутренние окна Swing. Самые важные функции многодокументного приложения – расположение «активного» окна над другими, сворачивание окон, их перетаскивание – обеспечиваются механизмами многослойной панели. Основное преимущество от использования многослойной панели для всплывающих подсказок и меню – это ускорение их работы. Вместо создания для каждой подсказки или меню нового тяжеловесного окна, располагающегося над компонентом, в

котором возник запрос на вывод подсказки или меню, Swing создает быстрый легковесный компонент. Этот компонент размещается в достаточно высоком слое многослойной панели выше в стопке всех остальных компонентов и используется для вывода подсказки или меню.

Многослойная панель позволяет организовать неограниченное количество слоев. Структура JLayeredPane включает несколько стандартных слоев, которые и используются всеми компонентами Swing, что позволяет обеспечить правильную работу всех механизмов многослойной панели. Стандартные слои:

- Default – слой используется для размещения всех обычных компонентов, которые добавляются в контейнер. В этом слое располагаются внутренние окна многодокументных приложений.
- Palette – слой предназначен для размещения окон с набором инструментов, которые обычно перекрывают остальные элементы интерфейса. Создавать такие окна позволяет панель JDesktopPane, которая размещает их в этом слое.
- Modal – слой планировался для размещения легковесных модальных диалоговых окон. Однако такие диалоговые окна пока не реализованы, так что этот слой в Swing в настоящее время не используется.
- Popup – слой наиболее часто используемый слой, служащий для размещения всплывающих меню и подсказок.
- Drag - самый верхний слой. Предназначен для операций перетаскивания (drag and drop), которые должны быть хорошо видны в интерфейсе программы.

Прозрачная панель JOptionPane

Прозрачная панель JOptionPane размещается корневой панелью выше всех элементов многослойной панели. За размещением JOptionPane следит корневая панель, которая размещает прозрачную панель выше многослойной панели, причем так, чтобы она полностью закрывала всю область окна, включая и область, занятую строкой меню.

JOptionPane используется в приложениях достаточно редко, поэтому по умолчанию корневая панель делает ее невидимой, что позволяет уменьшить нагрузку на систему рисования. Следует иметь в виду, что, если вы делаете прозрачную панель видимой, нужно быть уверенным в том, что она прозрачна (ее свойство opaque равно false), поскольку в противном случае она закроет все остальные элементы корневой панели, и остальной интерфейс будет невидим.

В каких случаях можно используется прозрачная панель JOptionPane. С ее помощью можно определять функции приложения, для реализации которых «с нуля» понадобились бы серьезные усилия. Прозрачную панель можно приспособить под автоматизированное тестирование пользовательского интерфейса. Синтезируемые в ней события позволяют отслеживать промежуточные отладочные результаты. Иногда такой подход гораздо эффективнее ручного тестирования.

Прозрачная панель JOptionPane может быть использована для эффектной анимации, «плавающей» поверх всех компонентов, включая строку меню, или для перехвата событий, если некоторые из них необходимо обрабатывать перед отправкой в основную часть пользовательского интерфейса.

Строка меню JMenuBar

Одной из важных особенностей использования корневой панели JRootPane в Swing, является необходимость размещения в окне строки меню JMenuBar. Серьезное приложение нельзя построить без какого-либо меню для получения доступа к функциям программы. Библиотека Swing предоставляет прекрасные возможности для создания удобных меню JMenuBar, которые также являются легковесными компонентами.

Строка меню JMenuBar размещается в многослойной панели в специальном слое FRAME_CONTENT_LAYER и занимает небольшое пространство в верхней части окна.

По размерам в длину строка меню равна размеру окна. Ширина строки меню зависит от содержащихся в ней компонентов.

Корневая панель следит, чтобы панель содержимого и строка меню JMenuBar не перекрывались. Если строка меню не требуется, то корневая панель использует все пространство для размещения панели содержимого.