

Частное учреждение образования  
Колледж бизнеса и права

УТВЕРЖДАЮ  
Заведующий  
методическим кабинетом  
\_\_\_\_\_ Е.В. Паскал  
«\_\_\_» \_\_\_\_\_ 2021

Специальность: 2-40 01 01 «Программное обеспечение информационных технологий»	Дисциплина: «Основы кроссплатформенного программирования»
---	---

ЛАБОРАТОРНАЯ РАБОТА № 14  
Инструкционно-технологическая карта

Тема: «Создание программ с графическим интерфейсом пользователя на языке Java. Классы пакета Swing»

Цель: Научиться работать с графическими интерфейсами пользователя, пользоваться пакетами Swing, классами пакета Swing.

Время выполнения: 6 часов

СОДЕРЖАНИЕ РАБОТЫ

1. Контрольные вопросы.
2. Теоретические сведения для выполнения работы.
3. Порядок выполнения работы.
4. Домашнее задание.
5. Литература.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ ДЛЯ ВЫПОЛНЕНИЯ РАБОТЫ

Для создания графического интерфейса пользователя в языке Java есть несколько графических пакетов библиотек (AWT, Swing, SWT и др.). Пакет Swing является одним из наиболее простых в применении и содержит классы для реализации большинства современных элементов GUI (Graphical User Interface – графический интерфейс пользователя).

Базовым объектом при создании пользовательского интерфейса является окно. В классификации языка Java им является класс JFrame. Для создания окна достаточно следующего кода:

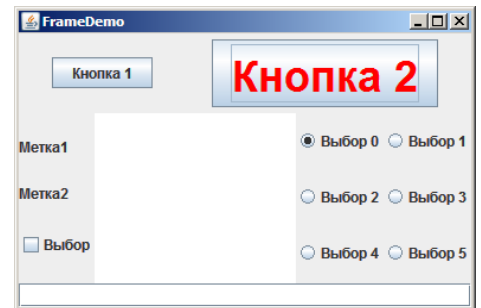
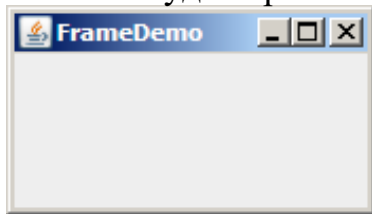
```
//файл MyFrame.java import javax.swing.*;  
public class MyFrame {  
    public static void main(String s[]) {
```

```

JFrame frame = new JFrame("FrameDemo"); // создаем окно с заголовком
FrameDemo frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE); // делаем базовым
действием при // закрытии окна выход из приложения, иначе окно закроется, а программа
в памяти останется frame.setSize(175,100); // задаем размер окна
frame.setVisible(true); // делаем
его видимым }
}

```

Результатом будет простое окно:



Дальнейшая работа с окном предусматривает использование менеджеров компоновки или компоновщиков. Это специальные классы, которые позволяют упаковывать содержимое окна и задавать нужное поведение всех элементов окна в зависимости от изменения размеров окна. В языке Java существует несколько компоновщиков:

**BorderLayout** – размещает элементы в один из пяти регионов (см. рис. 5), указанный при добавлении элемента в контейнер: *наверх*, *вниз*, *влево*, *вправо*, *в центр* (или *север*, *юг*, *запад*, *восток*, *центр*); при этом, если в какой-либо регион не был добавлен элемент, то этот регион не отображается в окне;

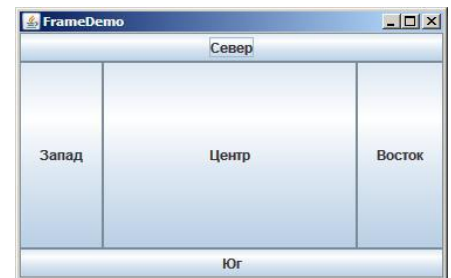


Рис. 5.  
BorderLayout.

**FlowLayout** – размещает элементы по порядку в том же направлении, что и ориентация контейнера (слева направо по умолчанию), применяя один из пяти видов выравнивания, указанного при создании менеджера (по центру по умолчанию). Данный менеджер используется по умолчанию в большинстве контейнерах;

**GridLayout** – размещает элементы таблично. Количество столбцов и строк указывается при создании менеджера. По умолчанию одна строка, а число столбцов равно числу элементов;

**BoxLayout** – размещает элементы по вертикали или по горизонтали. Обычно он используется не напрямую, а через контейнерный класс **Box**, который имеет дополнительные возможности;

**CardLayout** – размещает элементы в виде колоды карт, т.е. в текущий момент может быть активным и видимым только один элемент (который лежит наверху колоды).

Рекомендуется в качестве основного компоновщика для созданного окна использовать **BorderLayout**, т.к. это позволяет максимально удобно использовать все доступное пространство формы: в верхней части можно создать меню и панели инструментов, в нижней – строку состояния, в

центральной – основные рабочие элементы, а боковые области использовать для дополнительных элементов приложения. Все элементы графического интерфейса пользователя можно располагать на специальных панелях с нужным компоновщиком, а эти панели уже помещать в нужные области окна (верх, низ и т.д.).

Таким образом, необходимо рассмотреть работу с панелями и базовыми элементами пользовательского интерфейса.

**JPanel** – класс для работы с панелями. Панель – это своеобразный контейнер, к которому можно применить нужный компоновщик и поместить туда необходимые элементы, причем саму панель видно не будет, если только не установить специальные параметры.

**JButton** – кнопка.

**JCheckBox** – кнопка-флажок; **JComboBox** – выпадающий список; **JLabel** – метка, надпись;

**JList** – список;

**JPasswordField** – текстовое поле для скрытого ввода;

**JProgressBar** – компонент для отображения числа в некотором диапазоне;

**JRadioButton** – переключатели, радио-кнопки, обычно используются с компонентом **ButtonGroup**;

**JSlider** – компонент, позволяющий выбрать значение из заданного диапазона;

**JSpinner** – компонент, позволяющий выбрать значение из указанной последовательности; **JTable** – таблица;

**TextField** – однострочное текстовое поле; **TextArea** – многострочное текстовое поле;

**JTree** – дерево элементов (иерархически упорядоченных). Например, программа, создающая окно следующего вида:

```
import java.awt.*;
import java.util.ArrayList; import javax.swing.*;
public class MyFrame {
    public static void main(String[] args) {
        JFrame frame = new JFrame("FrameDemo");// создаем окно с заголовком FrameDemo
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);//делаем базовым
        действием при //закрытие окна выход из приложения, иначе окно закроется, а
        программа в памяти останется frame.setSize(500,400);//задаем размер окна
        JPanel myPanel1=new JPanel();//создаем панель
        myPanel1.setLayout(new FlowLayout());//устанавливаем для нее
        компоновщик myPanel1.add(new JButton("Кнопка 1"));//добавляем кнопку
        //вторую кнопку делаем по-другому, отдельным объектом, причем оформляем ее
        через html-код
        JButton myButton2 = new JButton("<html><b><font color=\"red\" size=14>Кнопка
        2</font></b></html>");//создаем распорку, которая будет стоять между кнопками
        Component horizontalStrut = Box.createHorizontalStrut(40);//с расстоянием в 40
```

точек

```

myPanel1.add(horizontalStrut); //добавляем распорку
myPanel1.add(myButton2); //добавляем вторую кнопку
Box myBox1=new Box(BoxLayout.Y_AXIS); //создаем объект Box для
компоновки BoxLayout
//с расположением объектов по вертикали (BoxLayout.Y_AXIS)
myBox1.add(Box.createVerticalStrut(20)); //добавляем распорку от верхнего края
окна
myBox1.add(new JLabel("Метка1"));
//добавляем метку
myBox1.add(Box.createVerticalGlue());
//добавляем пружину – она будет увеличиваться с
//увеличением окна и уменьшаться с уменьшением окна, тем самым изменяя
расстояние между объектами myBox1.add(new JLabel("Метка2")); //добавляем еще одну
метку myBox1.add(Box.createVerticalGlue());
//добавляем еще одну пружину
myBox1.add(new JCheckBox("Выбор")); //добавляем чекбокс
myBox1.add(Box.createVerticalStrut(20));
// добавляем распорку от нижнего края окна
ButtonGroup myGroup=new ButtonGroup(); //создаем группу, в которой будут
радиокнопки JPanel myPanel2=new JPanel();
// создаем панель для радиокнопок
//Создаем массив радиокнопок
ArrayList<JRadioButton> masRB=new ArrayList<JRadioButton>();
myPanel2.setLayout(new GridLayout(3,2)); // устанавливаем компоновщик для
табличного
//размещения объектов в 3 строки и 2 столбца
//в цикле будем добавлять радиокнопки и в массив, и в группу, и на
панель
for (int i=0;i<6;i++){
masRB.add(new JRadioButton("Выбор "+i)); //добавляем радиокнопку в
массив
//masRB.get(i) возвращает i-ю радиокнопку
myGroup.add(masRB.get(i)); //вставляем ее в группу
myPanel2.add(masRB.get(i)); //и добавляем на панель
}
masRB.get(0).setSelected(true); //устанавливаем выбранной 0-ю
радиокнопку
//теперь можно добавить все на форму в нужные области компоновки
BorderLayout frame.add(myPanel1, BorderLayout.NORTH);
frame.add(myBox1, BorderLayout.WEST);
frame.add(new JTextArea(), BorderLayout.CENTER); // создаем текстовую область и
добавляем ее
//в центр окна frame.add(myPanel2, BorderLayout.EAST);
frame.add(new JTextField(), BorderLayout.SOUTH); // создаем текстовое поле и
добавляем его //в нижнюю область окна
frame.setVisible(true); // делаем окно видимым
frame.pack(); //упаковываем окно, чтобы привести его к оптимальному размеру, при
котором //все элементы видны
frame.setMinimumSize(frame.getSize()); // и делаем этот размер

```

минимальным }

}

Или можно создать отдельные методы для заполнения каждого из регионов окна:

```
import java.awt.*;
```

```
import java.util.ArrayList; import javax.swing.*;
```

```
public class MyFrame {
    public static void main(String[] args) {
        JFrame frame = new JFrame("FrameDemo");// создаем окно с заголовком FrameDemo
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE); frame.setSize(500,400);//задаем
        размер окна
        setNorth(frame); //вызываем метод для заполнения верхней области
        setWest(frame); //вызываем метод для заполнения левой области
        setEast(frame); //вызываем метод для заполнения правой области
        setCenter(frame); //вызываем метод для заполнения центральной области
        setSouth(frame); //вызываем метод для заполнения нижней области
        frame.setVisible(true);//делаем окно видимым
        frame.pack(); //упаковываем его
        frame.setMinimumSize(frame.getSize());
    }
    public static void setNorth(JFrame fr){ //метод для заполнения верхней области
        JPanel myPanel1=new JPanel();
        myPanel1.setLayout(new FlowLayout());
        myPanel1.add(new JButton("Кнопка 1"));
        JButton myButton2 = new JButton("<html><b><font color=\"red\" size=14>Кнопка
2</font></b></html>");
        Component horizontalStrut = Box.createHorizontalStrut(40);
        myPanel1.add(horizontalStrut);
        myPanel1.add(myButton2);
        fr.add(myPanel1,BorderLayout.NORTH);
    }
    public static void setWest(JFrame fr){ //метод для заполнения левой области
        Box myBox1=new Box(BoxLayout.Y_AXIS);
        myBox1.add(Box.createVerticalStrut(20));
        myBox1.add(new JLabel("Метка1"));
        myBox1.add(Box.createVerticalGlue());
        myBox1.add(new JLabel("Метка2"));
        myBox1.add(Box.createVerticalGlue());
        myBox1.add(new JCheckBox("Выбор"));
        myBox1.add(Box.createVerticalStrut(20));
        fr.add(myBox1,BorderLayout.WEST);
    }
    public static void setEast(JFrame fr){ //метод для заполнения правой области
        ButtonGroup myGroup=new ButtonGroup();
        JPanel myPanel2=new JPanel();
        ArrayList<JRadioButton> masRB=new ArrayList<JRadioButton>();
        myPanel2.setLayout(new GridLayout(3,2));
        for (int i=0;i<6;i++){
            masRB.add(new JRadioButton("Выбор "+i));
```

```

myGroup.add(masRB.get(i)); myPanel2.add(masRB.get(i));
} masRB.get(0).setSelected(true);
fr.add(myPanel2,BorderLayout.EAST); }
public static void setCenter(JFrame fr){ //метод для заполнения центральной области
fr.add(new JTextArea(),BorderLayout.CENTER);
}
public static void setSouth(JFrame fr){ //метод для заполнения нижней области
fr.add(new JTextField(),BorderLayout.SOUTH); } }

```

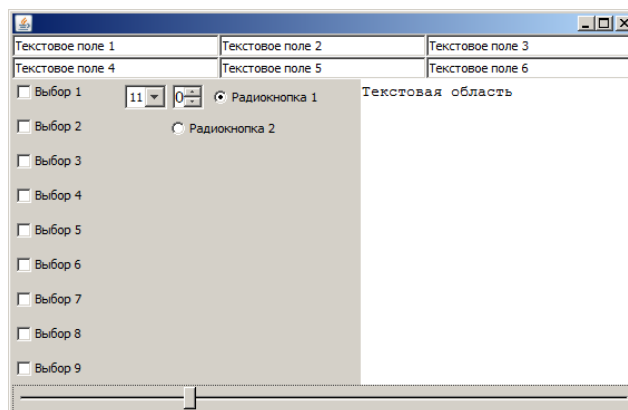
## КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Какие библиотеки используются для создания графического интерфейса.
2. Что такое компоновщик?
3. Перечислите немножко компоновщиков.
4. Охарактеризуйте GridLayout.
5. Охарактеризуйте BorderLayout.
6. Что такое панель? Какой класс используется для работы с панелью?

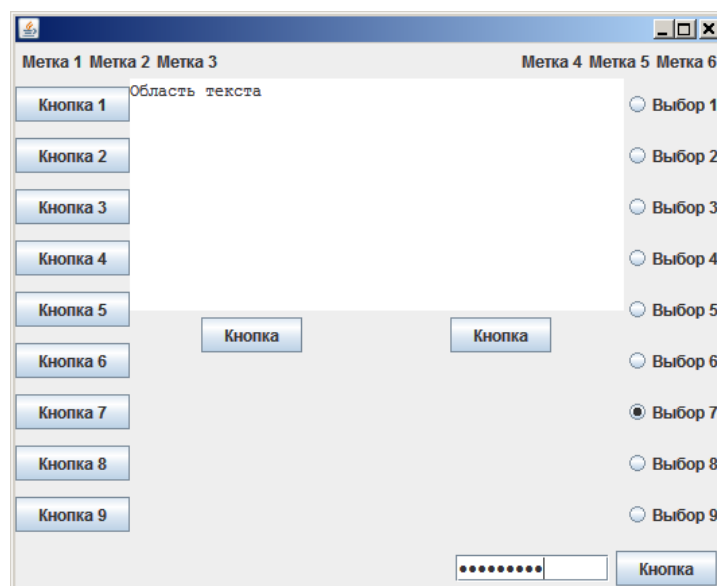
## ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Создать интерфейс по изображению (использовать разные виды компоновок, если одинаковых объектов больше 3, то использовать массив объектов):

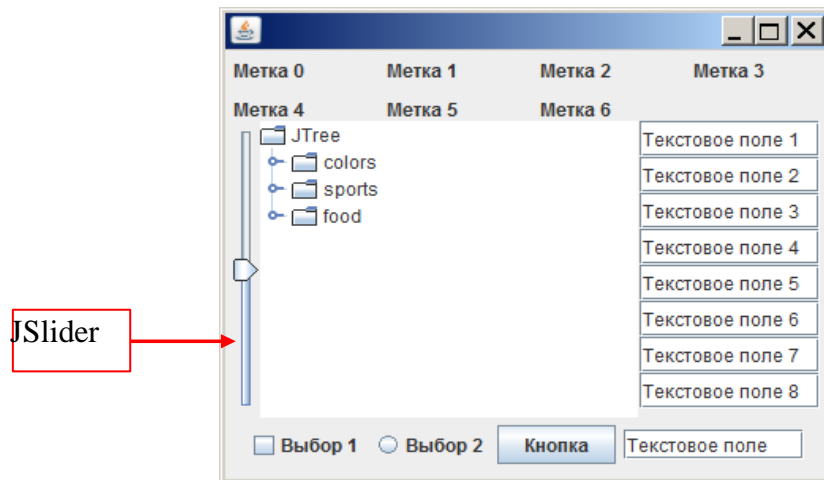
1.



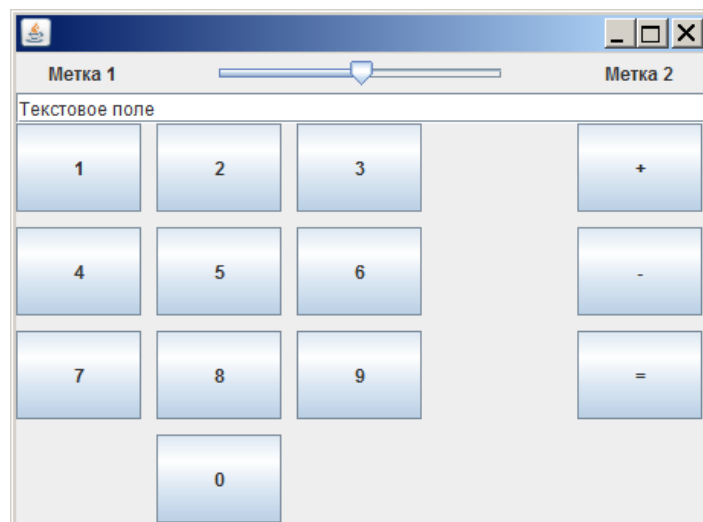
2.



3.



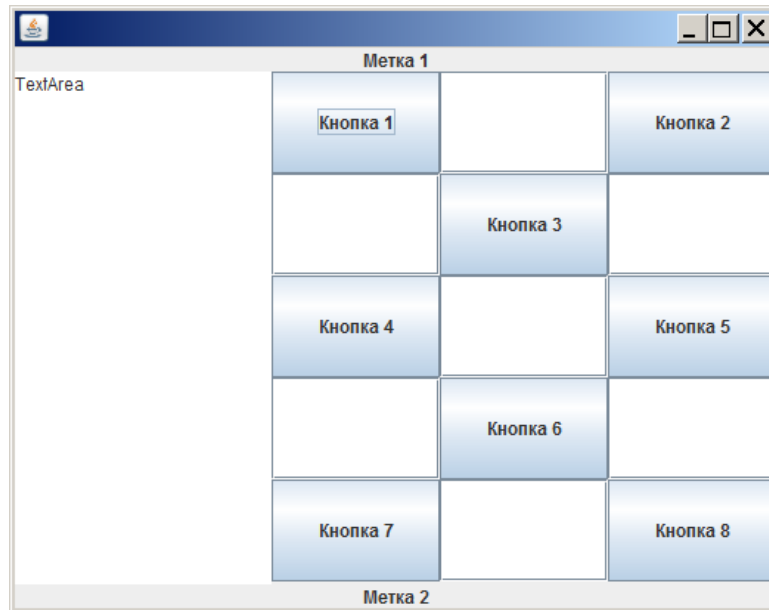
4.



5.



6.



## ДОМАШНЕЕ ЗАДАНИЕ

Индивидуальное задание

### ЛИТЕРАТУРА

И. Н. Блинов В. С. Романчик, Java, Четыре четверти, 2020.

Преподаватель

А.С.Кибисова

Рассмотрено на заседании цикловой комиссии  
программного обеспечения информационных  
технологий

Протокол № \_\_\_\_\_ от «\_\_\_» \_\_\_\_\_ 2021

Председатель ЦК \_\_\_\_\_ В.Ю.Михалевич