Разработка GUI на Java

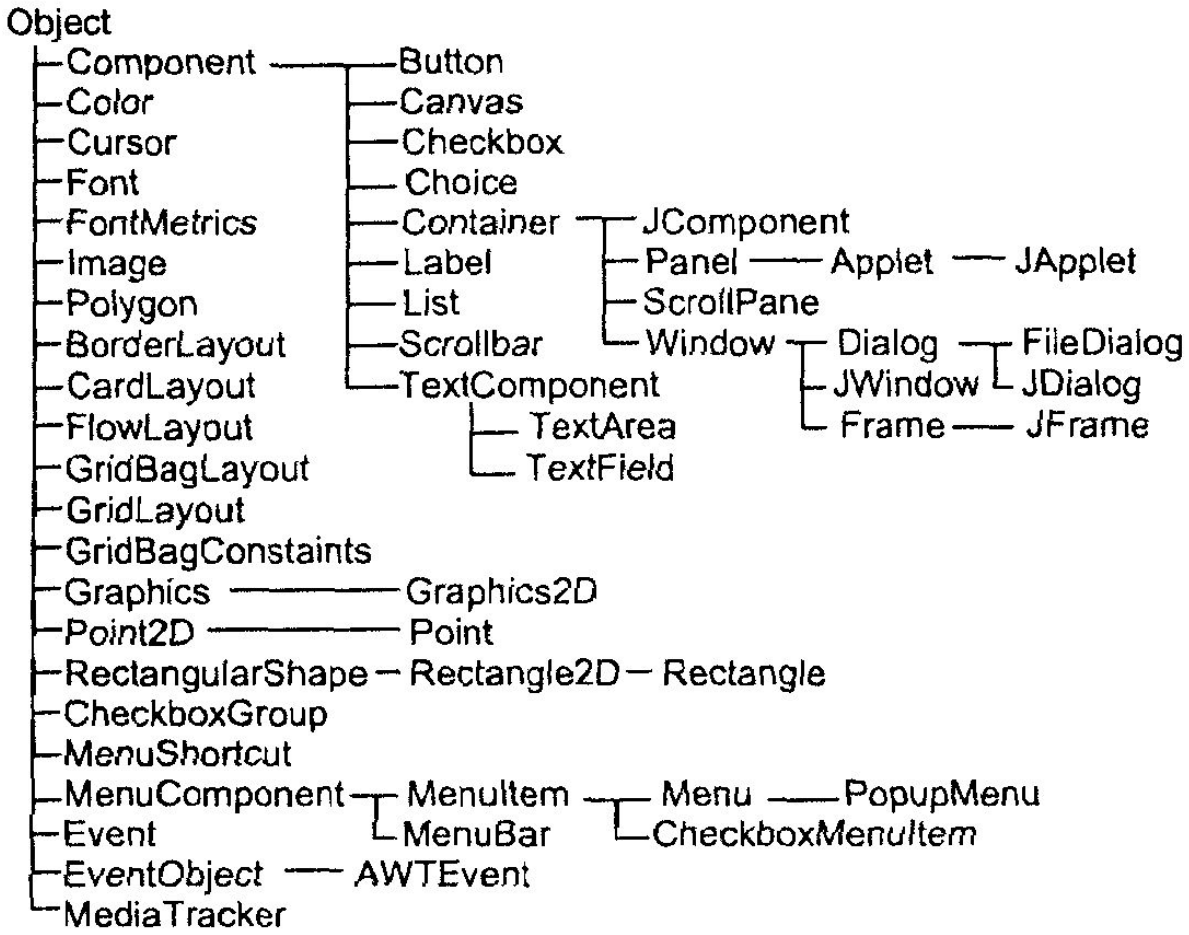
- библиотека AWT (Abstract Window Toolkit) - пакет java.awt.*
- библиотека JFC (Java Foundation Classes) - пакет javax.swing.*

Frame (AWT) – JFrame (Swing)

Button (AWT) – JButton (Swing)

…

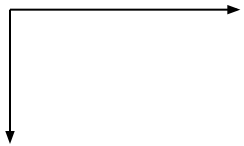
Классы для создания GUI



Класс Component (абстрактный)

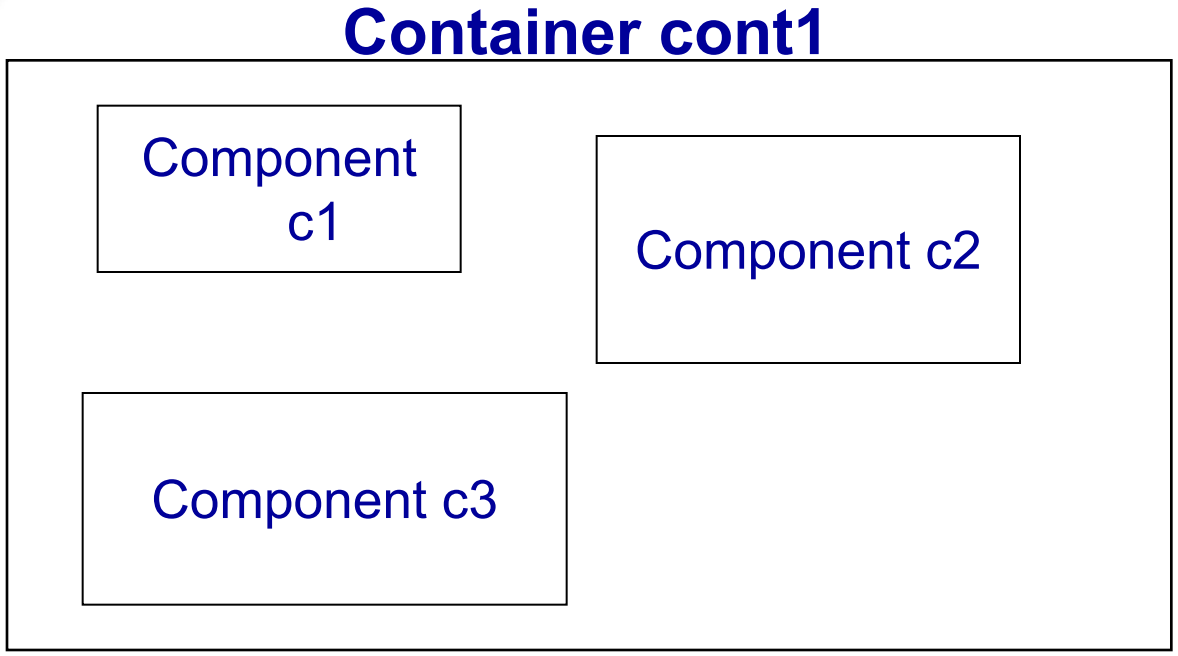
Ox

Oy



Dimension getSize()	Возвращает размеры компонента
setBounds (int x, int y, int width, int height)	Устанавливает размеры компонента
int getWidth() int getHeight()	Ширина компонента Высота компонента
isEnabled() setEnabled(boolean b)	Проверка доступности Установка доступности
isVisible() setVisible(boolean b)	Проверка видимости Установка видимости

Класс Container



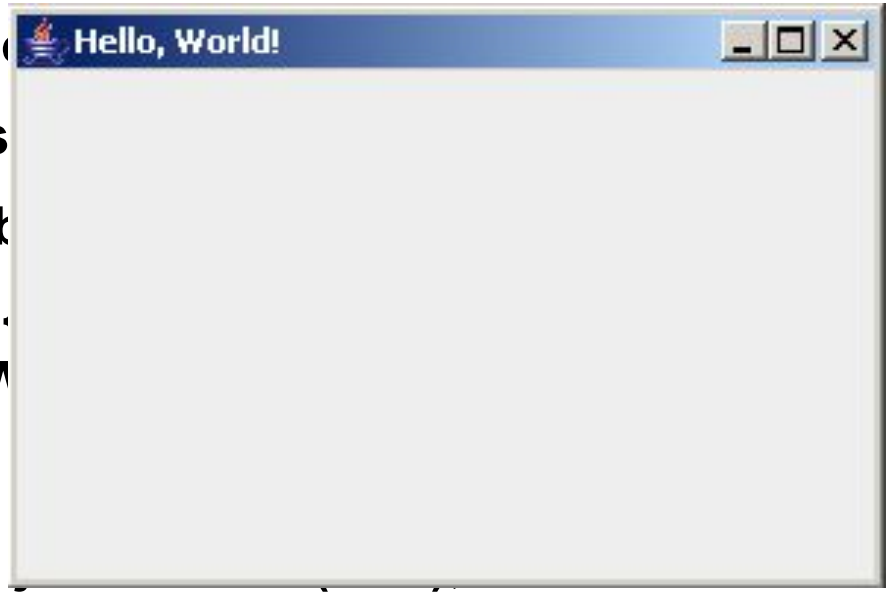
Добавление компонента к контейнеру: Component add (Component c)

Container является невидимым компонентом.

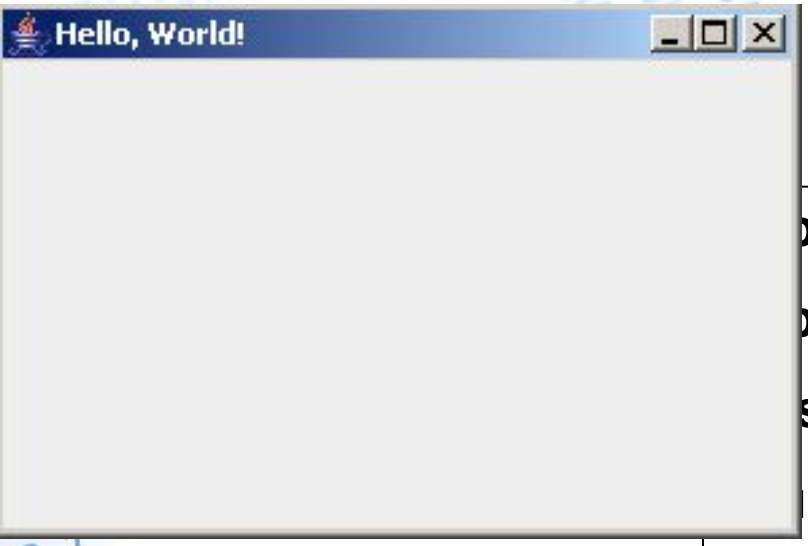
Все класс окон унаследованы от класса Container.

Классы окон. JFrame

```
import java.awt.*;
impo
clas
{pub
    W
    }
}
```



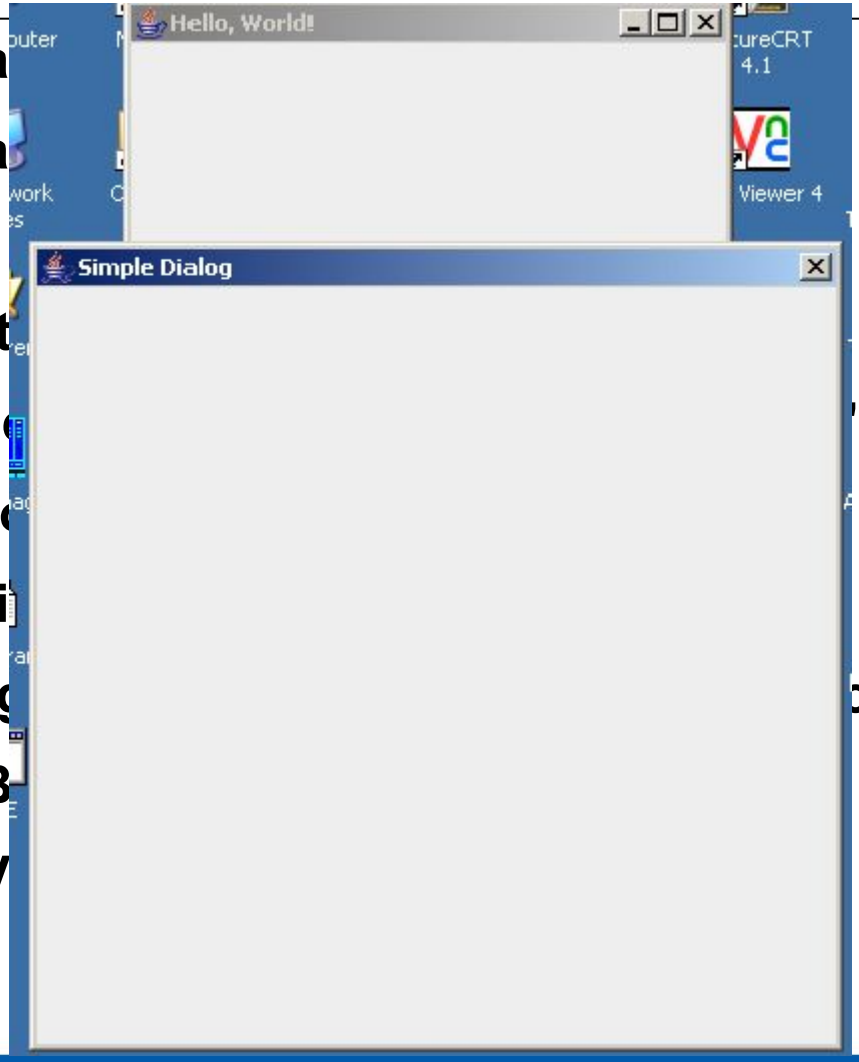
Классы окон. JWindow




```
ort java.awt.*;
ort javax.swing.*;
ss Test
blic static void main(String args[ ])
 {JFrame jf  = new JFrame("Hello,
World!");
unds (100,100,300,200);
ible(true);
w jw = new JWindow (jf);
ounds (400,400,500,200);
 jw.show(); }
}
```

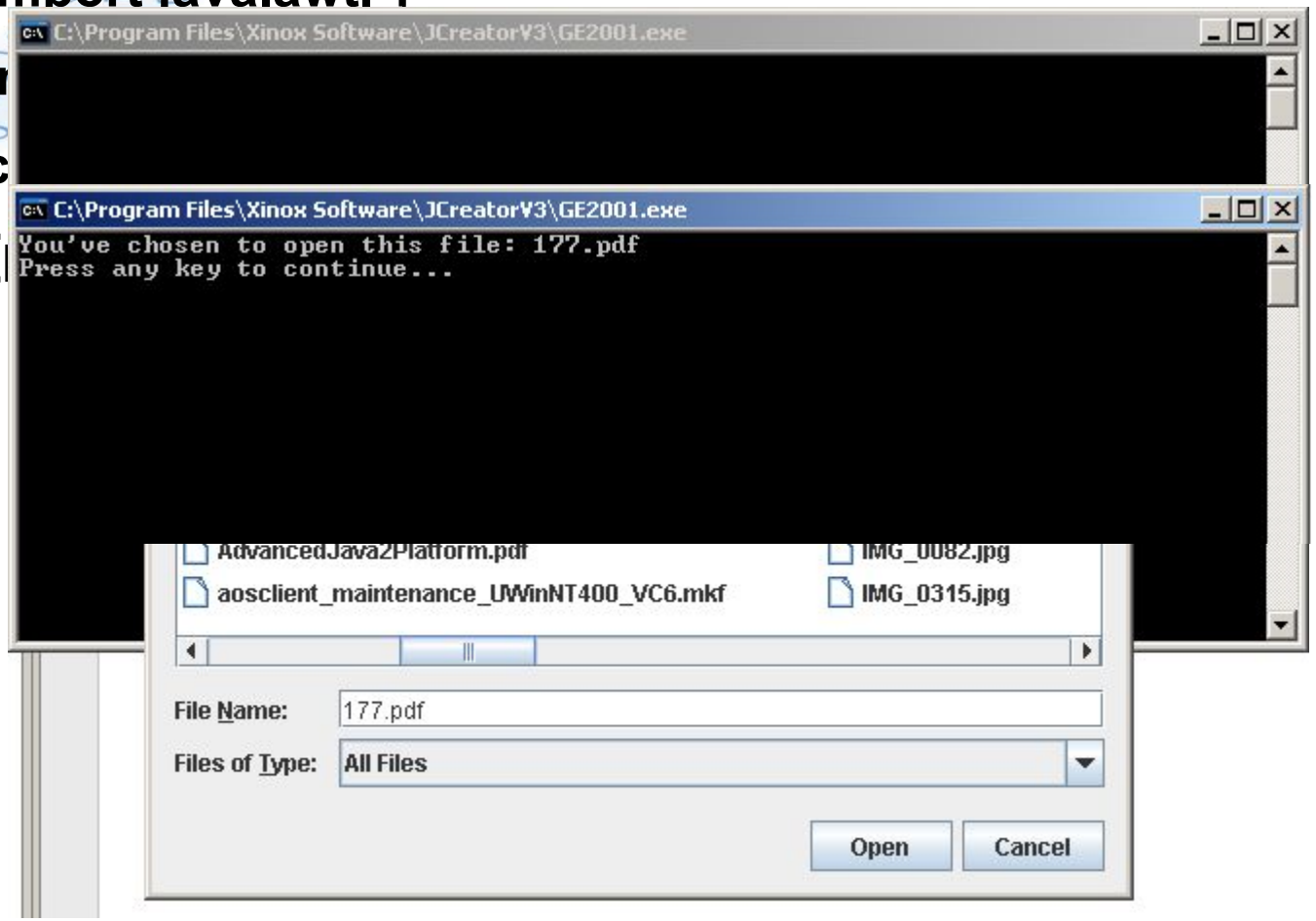
Классы окон. JDialog

```
import java
import java
class Test
{public stat
      {JFram                                    ");
       jf.setB
       jf.setVi
       JDialog                                  og",true);
       jd.setB
       jd.setV
}
```



Классы окон. JFileChooser

```
import java.awt.*;
i
c
{
}
```



Графический контекст

При создании компонента автоматически формируется его графический контекст (graphics context). Контекст содержит текущий и альтернативный цвет рисования и цвет фона, текущий шрифт для вывода текста, систему координат. Управляет контекстом класс Graphics. Т.к. графический контекст сильно зависит от конкретной графической платформы, этот класс является абстрактным. Каждая JVM реализует методы этих классов, создает их экземпляры для компонента. Получить ссылку на созданный графический контекст можно с помощью метода Graphics getGraphics().

Графический контекст. Методы

drawLine (int x1,int y1, int x2, int y2)	Линия из (x1,y1) в (x2,y2)
drawRect (int x, int y, int width, int height)	Прямоугольник: левый верхний угол (x,y), ширина width, высота height
draw3DRect (int x, int y, int width, int height, boolean raised)	3D-прямоугольник: rasied = true – «выпуклый» raised = false – «вдавленный»
drawOval (int x, int y, int width, int height)	Овал, вписанный в прямоугольник, заданный аргументами метода
drawArc (int x, int y, int width, int height, int startAngle, int arc)	Дуга овала, вписанного в прямоугольник. Величина дуги arc градусов. Отсчитывается от startAngle. Угол отсчитывается в градусах от оси Ox. Положительный угол - против часовой стрелки, отрицательный – по часовой стрелке.

Графический контекст. Методы

drawRoundRect (int x, int y, int width, int height, int ar-cWidth, int arcHeight)	Прямоугольник с закругленными краями. Закругления вычерчиваются четвертинками овалов, вписанных в прямоугольники шириной arcWidth и высотой arcHeight, построенные в углах основного прямоугольника.
drawPolyline (int xPoints[ ], int yPoints[ ], int nPoints)	Чертит ломаную линию с вершинами в точках (xPoints[i], yPoints[i]) и числом вершин nPoints
drawPolygon(int xPoints[ ], int yPoints[ ], int nPoints)	Чертит замкнутую ломаную с вершинами в точках (xPoints[i], yPoints[i]) и числом вершин nPoints
fillRect(int x, int y, int width, int height)	Эти методы аналогичны соответствующим методам draw…(), но в отличие от них вычерчивают фигуры, залитые текущим цветом
fillOval(int x, int y, int width, int height) и т.д.	

Графический контекст. Методы

drawString (String s, int x, int y)	Выводит строку s. Вывод начинается с позиции (x,y).
drawBytes (byte b[ ],int offset, int length, int x, int y)	Выводит length элементов массива b, начиная с ин-декса offset. Вывод начинается с позиции (x,y).
drawChars (char ch[ ], int offset, int length, int x, int y)	Выводит length элементов массива ch, начиная с индекса offset. Вывод начинается с позиции (x,y).

Графический контекст. Работа с цветом

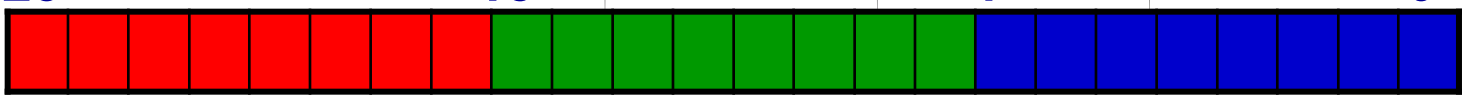
Color (int red, int green, int blue)

0-255 0-255 0-255

Color (float red, float green, float blue)

0.0 -1.0 0.0 -1.0 0.0 -1.0

Color (int rgb) 23 15 7 0



Color(int r, int g, int b, int a)

Прозрачность: 0 - полная, 255 – отсутств.

Графический контекст. Работа с цветом

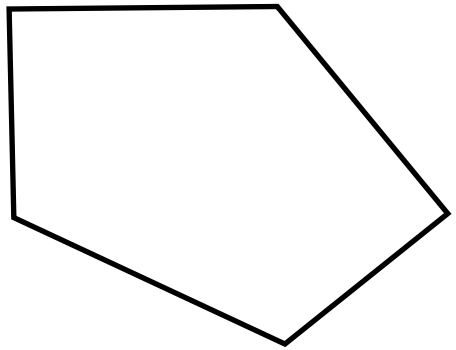
Цветовые константы:

Color.black Color.blue
Color.cyan Color.darkGray
Color.gray Color.green,
Color.lightGray Color.magenta
Color.orange Color.pink
Color.red Color.white,
Color.yellow

Графический контекст. Класс Polygon

Polygon(int xPoints[ ], int yPoints[ ], int nPoints)

Создается многоугольник с вершинами (xPoints[i], yPoints[i]). Число вершин – nPoints.



Графический контекст. Класс Polygon

boolean contains (double x, double y)	Проверяет, лежит ли точка (x,y) внутри многоугольника
boolean contains (double x, double y, double w, double h)	Проверяет, лежит ли прямоугольник, заданный аргументами метода внутри многоугольника
boolean contains (int x, int y)	Проверяет, лежит ли точка (x,y) внутри многоугольника
boolean contains (Point p)	Проверяет, лежит ли точка (x,y) внутри многоугольника
boolean intersects (double x, double y, double w, double h)	Проверяет, пересекается ли прямоугольник, заданный аргументами метода с многоугольником

Графический контекст. Элементы управления

- Текстовые метки • Кнопки
- Флажки • «Радиопереключатели»
- Списки • Поля со списком
- Текстовые поля • Текстовые области
- Панели прокрутки • Панели с вкладками
- Деревья • Таблицы

Добавление компонента к контейнеру: Component add (Component c)

Удаление компонента из контейнера: void remove (Component obj).

Графический контекст. Элементы управления

В библиотеке Swing компоненты перед выводом на экран должны быть помещены не в окно, а внутрь специального контейнера, который называется панель содержания (content pane). Этот контейнер не нужно создавать, его можно получить для окна подобно графическому контексту:

```
JFrame f = new JFrame("Test");
Container cp = f.getContentPane();
```

После этого все операции добавления/удаления компонентов выполняются для панели содержания, а не для самого окна.

Графический контекст. Элементы управления

Графические изображения

Конструктор класса ImageIcon:

ImageIcon(String filename)

Основные методы класса ImageIcon:

int getIconHeight()

int getIconWidth()

void paintIcon(Component comp, Graphics g, int x, int y)

Графический контекст. Элементы управления

Текстовые метки

JLabel (String str)

JLabel (Icon id_icon)

JLabel (String str, Icon id_icon, int align)

JLabel.LEFT - выравнивание по левому краю

JLabel.RIGHT - выравнивание по правому краю

JLabel.CENTER - выравнивание по центру

void setText(String str) String getText( )

void setAlignment(int align) int getAlignment( )

void setIcon(Icon id_icon) Icon getIcon( )

Графический контекст. Элементы управления

Управляющие кнопки

JButton(String str)

JButton(Icon id_icon)

Методы, позволяющие управлять видом кнопки:

String getLabel( ) void setLabel(String str)

Icon getLabel( ) void setIcon(Icon id_icon)

Графический контекст. Элементы управления

Флажки - переключатели

JCheckBox (String str, Icon id_icon, boolean state)

Управление состоянием флажка из программы осуществляется с помощью методов:

boolean getSelected()

void setSelected(boolean state)

Графический контекст. Элементы управления

«Радио-переключатели»

JRadioButton (String str, Icon id_icon, boolean state)

«Радиопереключатели» должны быть объединены в группу, в пределах которой будет обеспечиваться уникальность выбора элемента. Чтобы создать группу создается экземпляр объекта специального класса ButtonGroup. Далее все созданные радиопереключатели добавляются в созданную группу (что не отменяет необходимости их добавления к панели содержания).

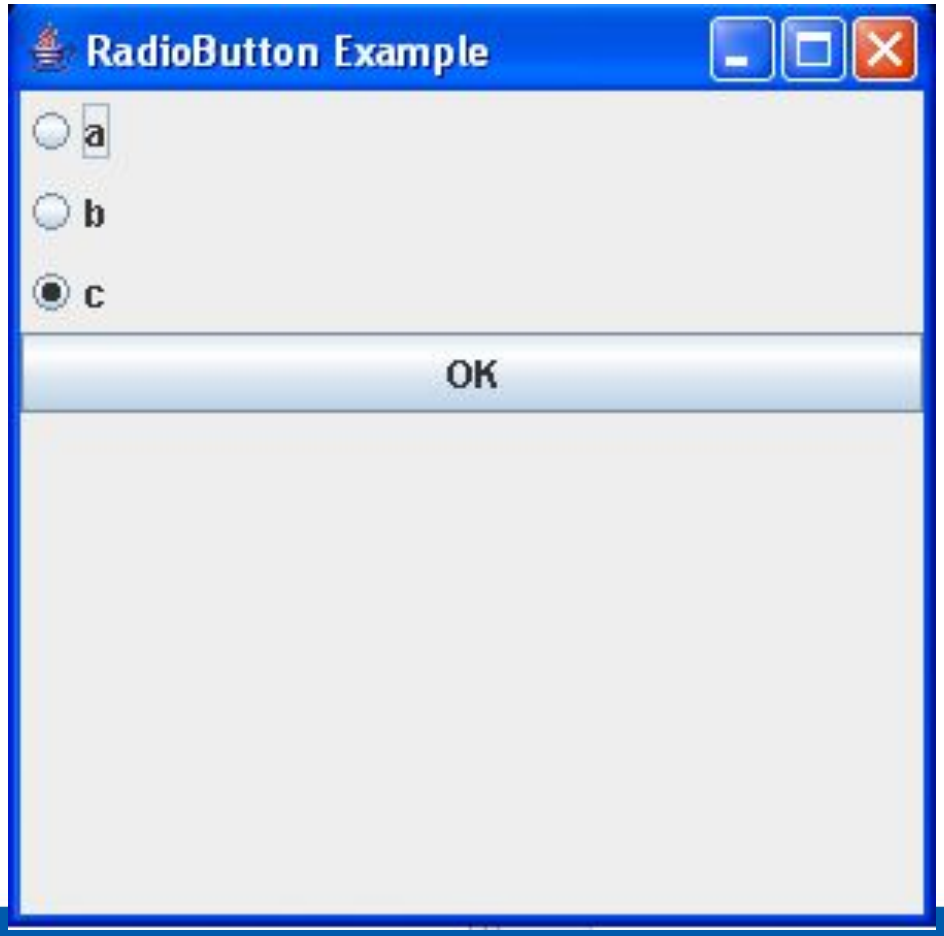
Графический контекст. Элементы управления

«Радио-переключатели»

```
JFrame f = new JFrame("RadioButton Example");
Container cp = f.getContentPane();
cp.setLayout(new GridLayout(5,1));
JRadioButton jrb1 = new JRadioButton("a",false);
JRadioButton jrb2 = new JRadioButton("b",false);
JRadioButton jrb3 = new JRadioButton("c",true);
ButtonGroup bg = new ButtonGroup();
JButton jb = new JButton("OK");
bg.add(jrb1);bg.add(jrb2);bg.add(jrb3);
cp.add(jrb1);cp.add(jrb2);cp.add(jrb3);cp.add(jb);
f.setVisible(true);
```

Графический контекст. Элементы управления

«Радио-переключатели»



Графический контекст. Элементы управления

Раскрывающиеся списки

JComboBox()

JComboBox(Vector v[ ])

JComboBox(Object obj[ ])

Добавление элемента в список: addItem(Object item)

Вставка элемента в список: insertItemAt(Object anObject, int index)

Удаление элементов из списка:

void removeItem (String text)

void removeItemAt (int index)

void RemoveAllItems ()

Графический контекст. Элементы управления

Раскрывающиеся списки

Работа с выбранными элементами:

String getSelectedItem( ) void setSelectedItem (Object obj)

int getSelectedIndex( ) void setSelectedIndex (int index)

Установка режима редактирования:

void setEditable (boolean state)

Графический контекст. Элементы управления

Списки

JList(Vector v[ ])

JList(Object obj[ ])

Объекты типа JList не поддерживают автоматическую прокрутку. Поэтому для того, чтобы элементы в списке могли прокручиваться, список необходимо поместить в специальный объект JScrollPane, а уже этот объект добавить в контейнер.

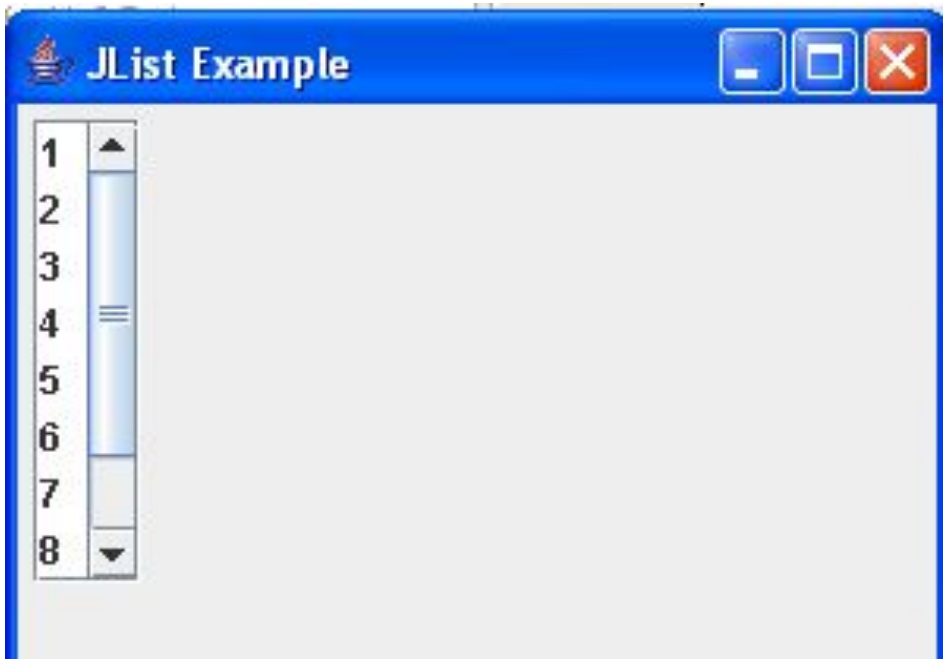
Графический контекст. Элементы управления

Списки

```
JFrame f = new JFrame();
Container cp = f.getContentPane();
f.setBounds(100,100,300,300);
String s[ ] = {"1","2","3","4","5","6","7","8","9","10"};
JList jl = new JList(s);
jl.setBounds(10,10,30,30);
JScrollPane p = new JScrollPane(jl);
cp.add(p);
f.setVisible(true);
```

Графический контекст. Элементы управления

Списки



Графический контекст. Элементы управления

Списки

Для определения выбранного элемента (элементов) используются методы:

```
Object getSelectedValue()
Object[ ] getSelectedValues()
int getSelectedIndex( )
int[ ] getSelectedIndices( )
```

Графический контекст. Элементы управления

Текстовые поля

JTextField( )

JTextField(int numChars)

JTextField(String str)

JTextField(String str, int numChars)

String getText()	Возвращает строку, содержащуюся в поле ввода.
void setText (String str)	Устанавливает в поле ввода текст str.
String getSelectedText()	Возвращает выделенную часть текста.
void select (int start, int end)	Программное выделение части текста в поле ввода начиная с позиции start до позиции end.
void setEchoChar(char ch)	Для ввода паролей

Графический контекст. Элементы управления

Текстовые области

JTextArea( )

JTextArea(int numLines, int numChars)

JTextArea(String str)

JTextArea(String str, int numLines, int numChars)

void append (String str)	Строка str добавляется к концу текущего текста
void insert (String str, int index)	Строка str вставляется в позицию, указанную в параметре index.
void replaceRange (String str, int start, int end)	Заменяет символы от start до end-1 строкой str.

Для обеспечения прокрутки JTextField и JTextArea должны быть помещены в контейнер JScrollPane.

Графический контекст. Элементы управления

Другие элементы

JTabbedPane – панель со вкладками

JTree – деревья

JTable – таблицы

JSplitPane – «расщепленные» окна

JSlider – ползунковый регулятор

JProgressBar – индикатор прогресса

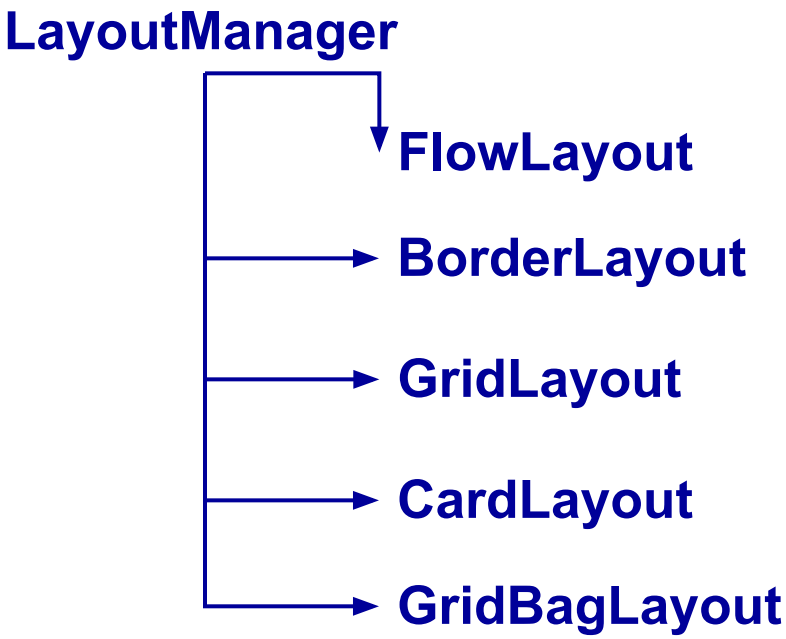
Менеджеры компоновки

Размещение компонентов в окне с помощью setBounds() имеет множество недостатков:

- необходимо вручную рассчитывать координаты
- при изменении размеров окна, компоненты «поплывут»
- теряется кроссплатформность

Поэтому существуют менеджеры компоновки – средства автоматического размещения компонентов в окне

Менеджеры компоновки



Менеджер компоновки устанавливается методом

void setLayout (LayoutManager obj)

Менеджеры компоновки FlowLayout

FlowLayout – менеджер поточной компоновки. Он размещает компоненты в окне начиная от левого верхнего угла слева направо и сверху вниз.

Конструкторы:

- **FlowLayout (int align, int h, int v)**
- **FlowLayout (int align)**
- **FlowLayout ()**

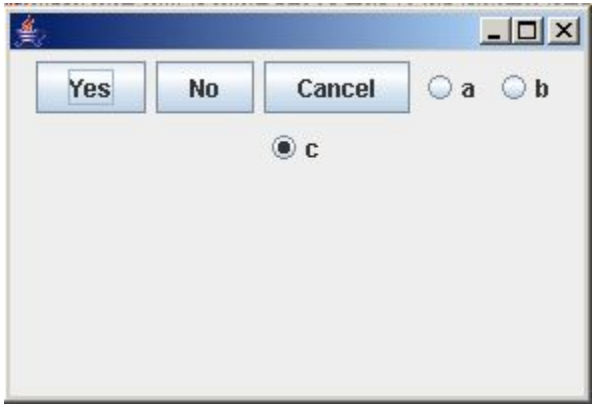
Методы:

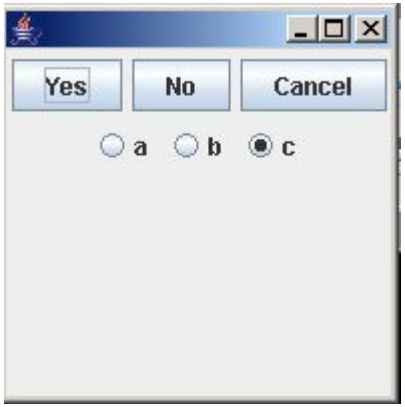
- **void setAlignment (int align)**
- **void setHgap (int hgap)**
- **void setVgap (int vgap)**

Менеджеры компоновки
FlowLayout

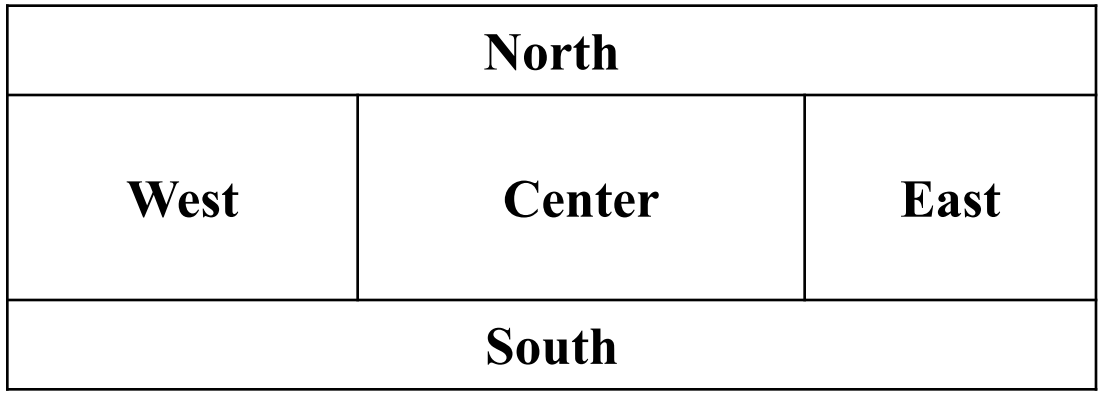
```
f.setLayout(new FlowLayout());
JRadioButton jrb1 = new JRadioButton("a",false);
JRadioButton jrb2 = new JRadioButton("b",false);
JRadioButton jrb3 = new JRadioButton("c",true);
JButton jb1 = new JButton ("Yes");
JButton jb2 = new JButton ("No");
JButton jb3 = new JButton ("Cancel");
ButtonGroup bg = new ButtonGroup();
bg.add(jrb1); bg.add(jrb2); bg.add(jrb3);
cp.add(jb1);cp.add(jb2);cp.add(jb3);cp.add(jrb1);cp.add(jrb2);
cp.add(jrb3);f.setVisible(true);
```

Менеджеры компоновки
FlowLayout





Менеджеры компоновки BorderLayout



Конструкторы класса BorderLayout:

BorderLayout ()

BorderLayout (int horz, int vert)

Менеджеры компоновки BorderLayout

Добавлять компоненты в этом случае можно с помощью специальной формы метода add():

void add (Component comp, Object region)

где параметр region должен принимать одно из следующих значений:

BorderLayout.NORTH

BorderLayout.SOUTH

BorderLayout.EAST

BorderLayout.WEST

BorderLayout.CENTER

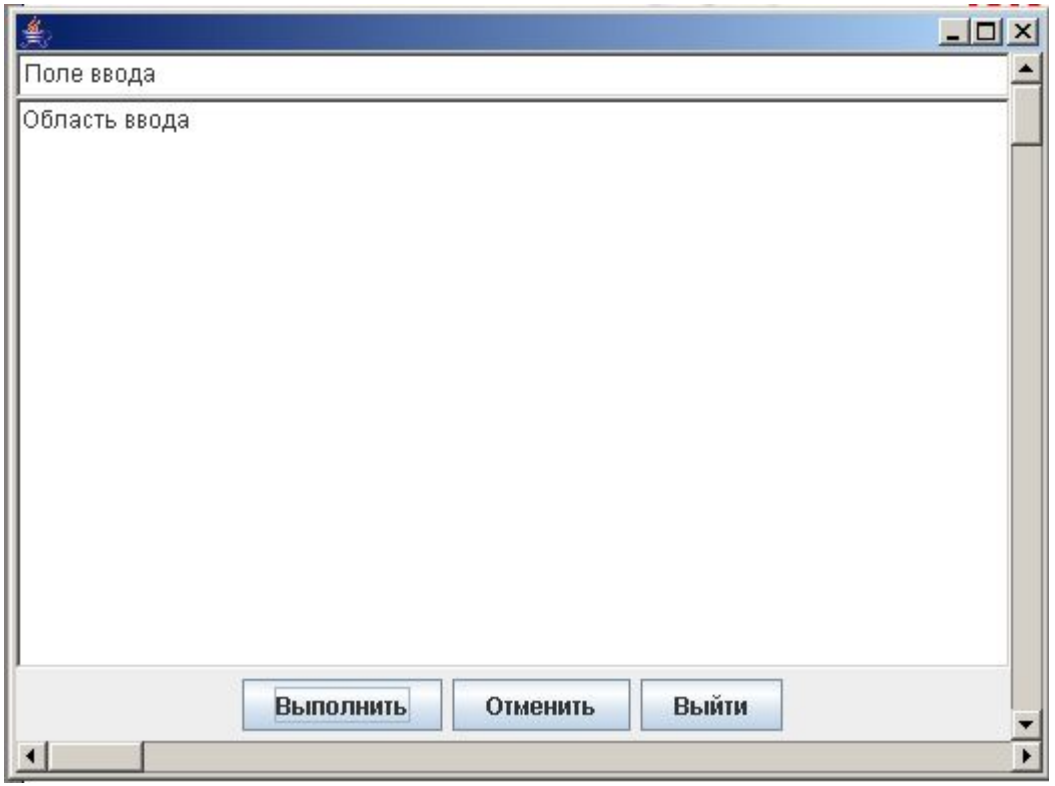
Менеджеры компоновки BorderLayout

```
JFrame f = new JFrame();
Container cp = f.getContentPane();
// Создаем панель p2 с тремя кнопками
Panel p2 = new Panel();
p2.add(new JButton(“Выполнить”));
p2.add(new JButton(“Отменить")) ;
p2.add(new JButton(“Выйти"));
// Создаем панель p1 и устанавливаем для нее
    компоновку BorderLayout
Panel p1 = new Panel();
p1.setLayout(new BorderLayout());
```

Менеджеры компоновки BorderLayout

```
p1.add (p2, BorderLayout.SOUTH);
p1.add(new TextField(«Поле ввода», 20), BorderLayout.NORTH);
p1.add(new TextArea («Область ввода», 5, 20,
    TextArea.SCROLLBARS_NONE), BorderLayout.CENTER);
cp.add (new Scrollbar (Scrollbar. HORIZONTAL),
    BorderLayout.SOUTH);
cp.add(new Scrollbar(Scrollbar.VERTICAL), BorderLayout.EAST) ;
cp.add(p1, BorderLayout.CENTER) ;
f.setVisible(true);
```

Менеджеры компоновки BorderLayout



Менеджеры компоновки GridLayout

Менеджер GridLayout размещает компоненты в двумерной сетке (иногда такое размещение называют табличным). Число строк и столбцов сетки следует задавать при создании экземпляра объекта GridLayout. Компоненты размещаются слева направо по строкам созданной таблицы в том порядке, в котором они заданы методом add().

Конструкторы, определенные в классе GridLayout:

GridLayout()

GridLayout(int numRows, int numColumns)

GridLayout(int numRows, int numColumns, int horz, int vert)

Менеджеры компоновки GridLayout

Внимание! Если число компонентов > числа клеток, то:

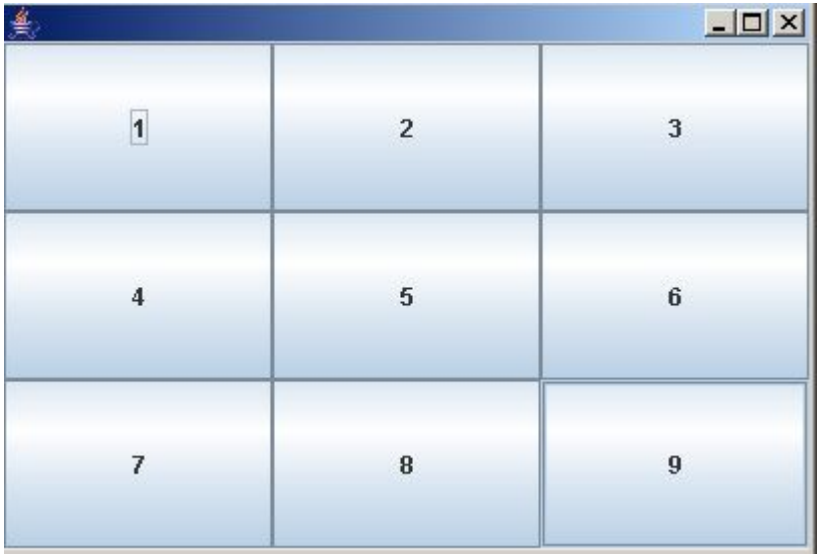
- если numRows != 0 & numColumns !=0 -> numColumns игнорируется и число столбцов устанавливается так, чтобы уместились все компоненты

- если numRows == 0 -> число строк устанавливается так, чтобы уместились все компоненты

Менеджеры компоновки GridLayout

```
JFrame f = new JFrame();
Container cp = f.getContentPane();
f.setLayout(new GridLayout(3, 2));
f.add(new JButton("1")); f.add(new JButton("2"));
f.add(new JButton("3")); f.add(new JButton("4"));
f.add(new JButton("5")); f.add(new JButton("6"));
f.add(new JButton("7")); f.add(new JButton("8"));
f.add(new JButton("9"));
f.setVisible(true);
```

Менеджеры компоновки GridLayout



Менеджеры компоновки CardLayout

Менеджер CardLayout показывает в контейнере только первый компонент. Остальные компоненты лежат под первым в определенном порядке как игральные карты в колоде.

Их расположение определяется порядком вызова методов add().

следующий компонент next(Container c)

предыдущий previous(Container c)

первый first(Container c)

последний last(Container c)

Аргумент этих методов – ссылка на контейнер, в который помещены компоненты, обычно this.

Менеджеры компоновки CardLayout

Конструкторы:

CardLayout() не отделяет компонент от границ контейнера

CardLayout(int hgap, int vgap) задает горизонтальные и вертикальные промежутки между компонентом и границами контейнера.

CardLayout позволяет организовать и произвольный доступ к компонентам. Метод add() для этого менеджера имеет вид

add(Component comp, Object name)

Показать нужный компонент с именем name можно с помощью метода

show(Container parent, String name)

Менеджеры компоновки CardLayout

```
Panel p = new Panel();
CardLayout c = new CardLayout();
p.setLayout(c);
Panel p1 = new Panel();
Panel p2 = new Panel();
Panel p3 = new Panel();
p.add(p1,"1");
p.add(p2,"2");
p.add(p3,"3");
c.next(p);
```

VE – Visual Editor

Разработка GUI с помощью визуального редактора

VE – Visual Editor. Плагин для среды разработки Eclipse IDE.

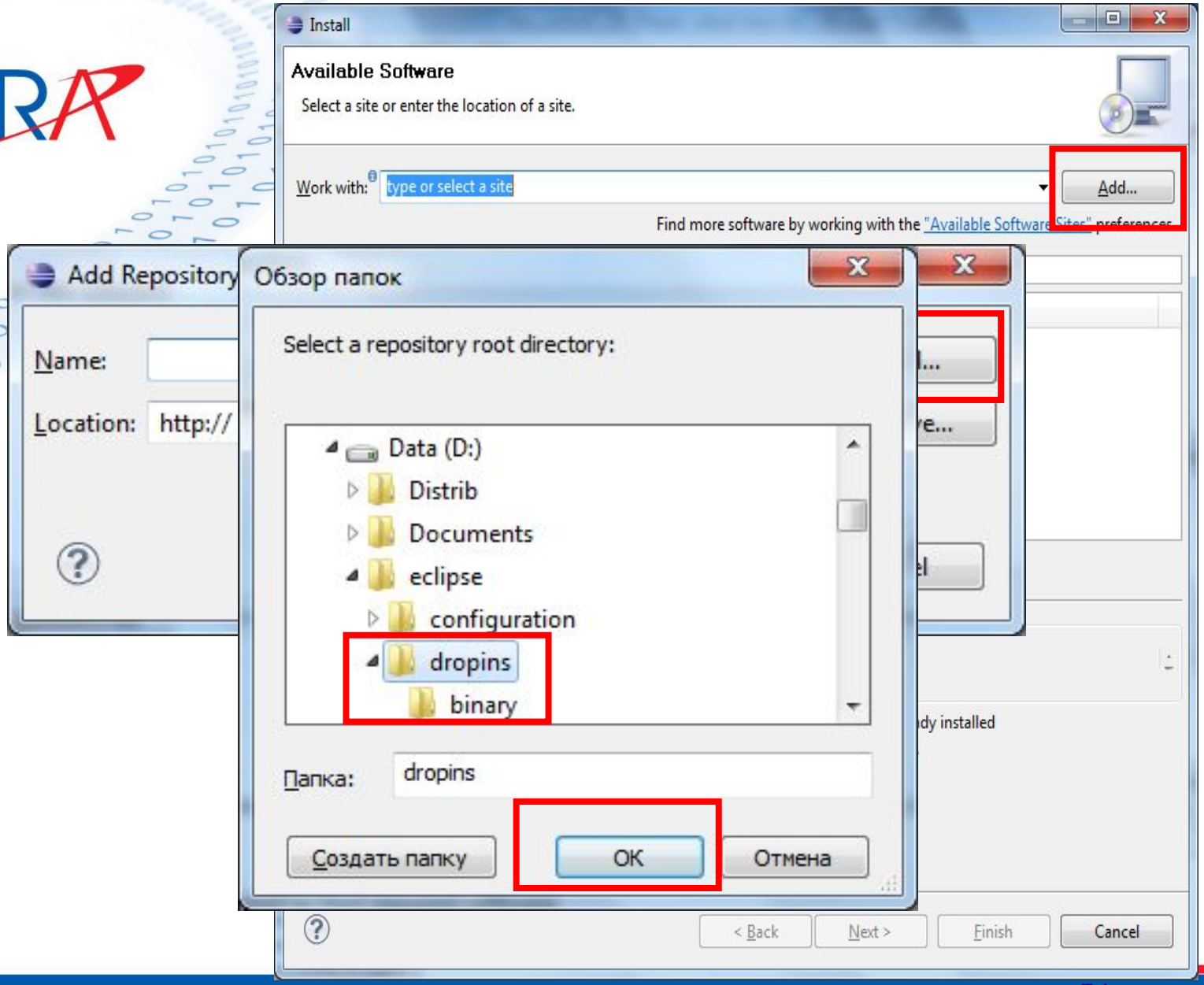
- **Позволяет работать с графическими библиотеками Java Swing и Java AWT.**
- **Упрощает работу с компонентами**
- **Поддерживает менеджеры компоновки**
- **Автоматическое добавление обработчиков (слушателей) событий.**

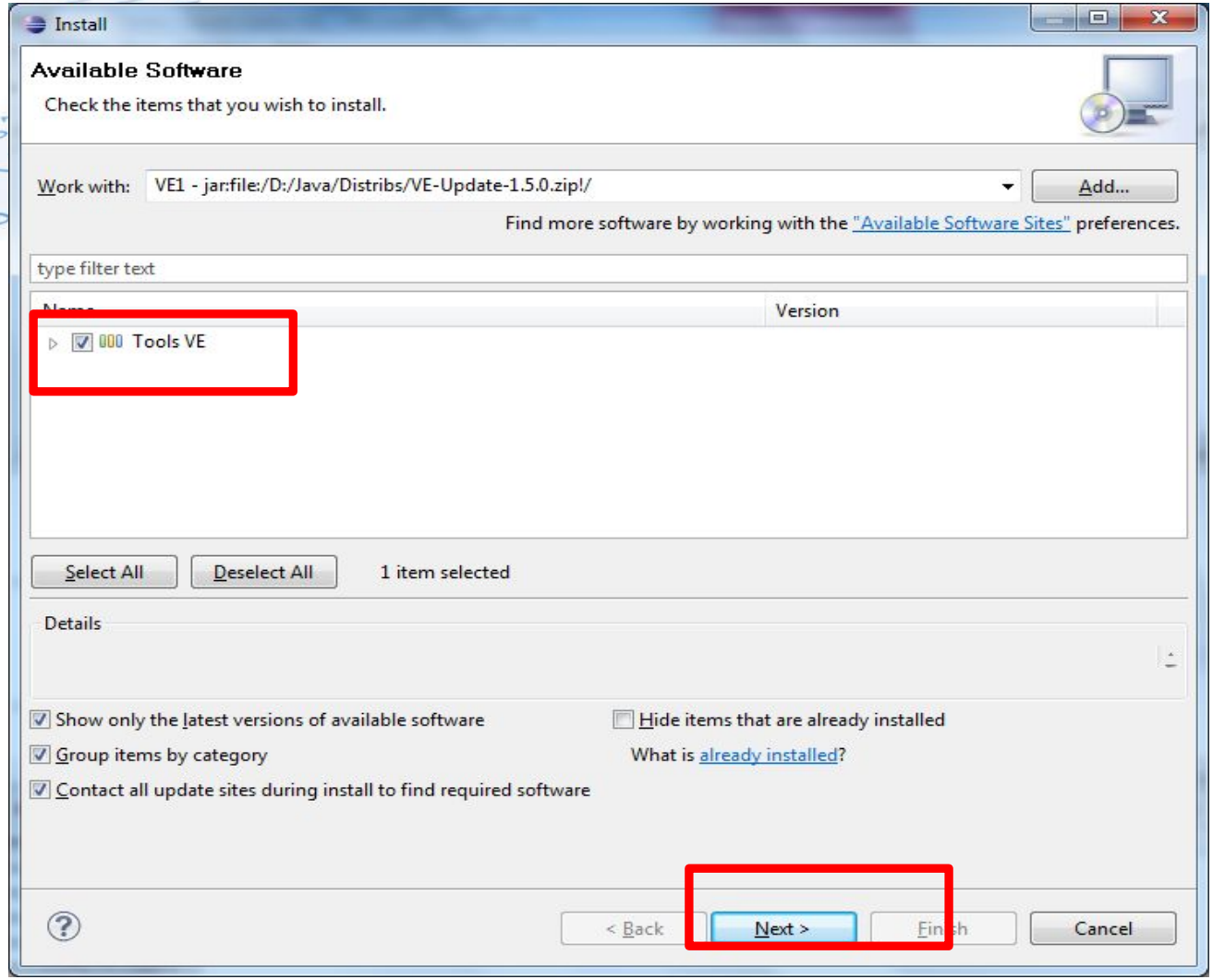
VE – Visual Editor

Установка

Зайти в Eclipse – Help->Install New Software.
Нажать кнопку Add и указать путь

http://visualeditor.sourceforge.net/updates/1.5.0

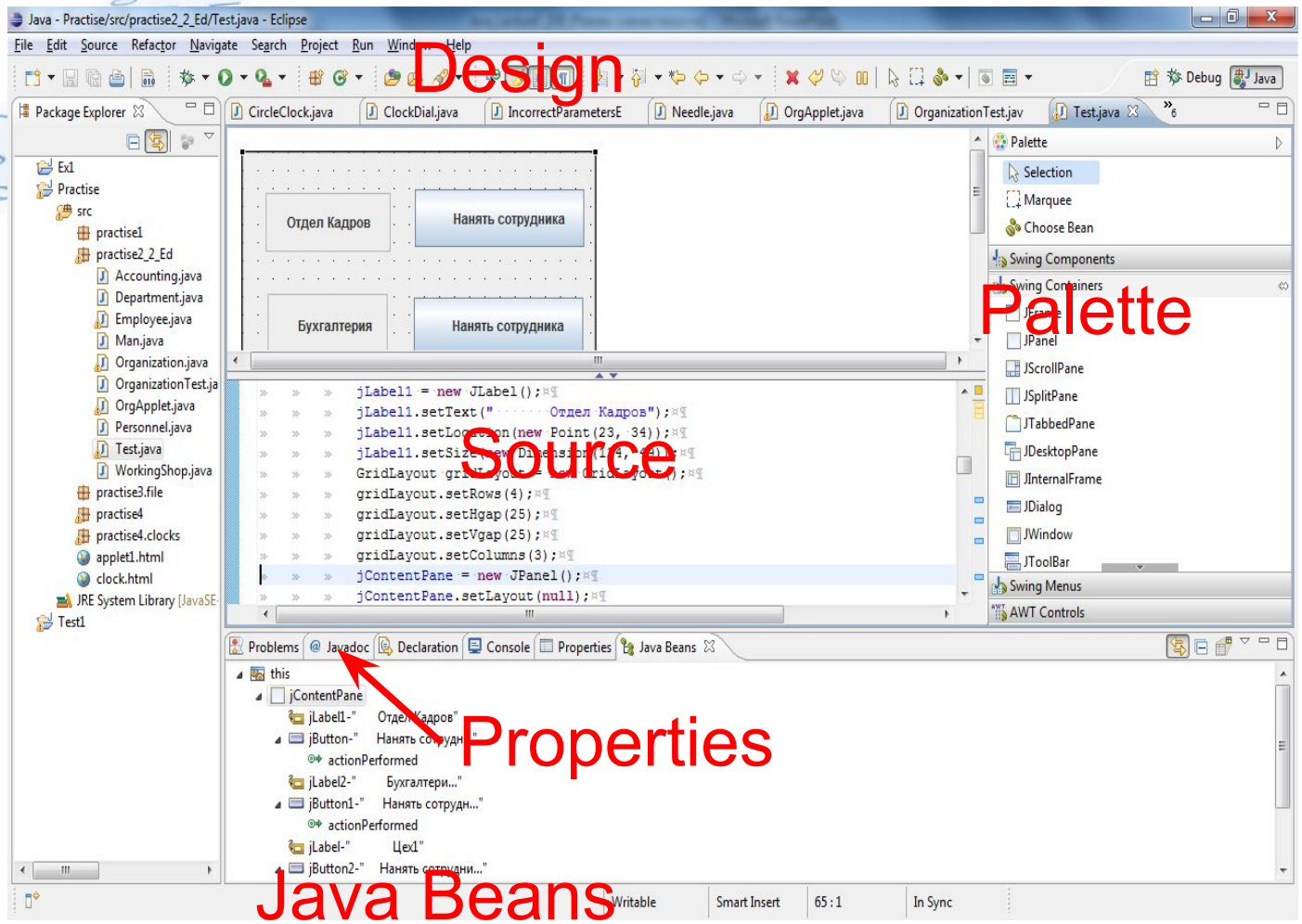

Install
Available Software
Select a site or enter the location of a site.
Work with:
type or select a site
Add...
Find more software by working with the "Available Software Sites" preferences
Add Repository
Name:
Location:
http://
Обзор папок
Select a repository root directory:
Data (D:)
Distrib
Documents
eclipse
configuration
dropins
binary
Папка:
dropins
Создать папку
OK
Отмена
< Back
Next >
Finish
Cancel


Install
Available Software
Check the items that you wish to install.
Work with:
VE1 - jar:file:/D:/Java/Distribs/VE-Update-1.5.0.zip!/
Add...
Find more software by working with the "Available Software Sites" preferences.
type filter text
Name
Version
Tools VE
Select All
Deselect All
1 item selected
Details
Show only the latest versions of available software
Group items by category
Contact all update sites during install to find required software
Hide items that are already installed
What is already installed?
< Back
Next >
Finish
Cancel

VE – Visual Editor

Три представления

- Design – как будет выглядеть окно
- Source - исходный код
- Palette - палитра графических элементов
- Properties – просмотр и установка свойств компонента.
- Java Beans – дерево вложенности компонентов


Java - Practise/src/practise2_2_Ed/Test.java - Eclipse
Design
Palette
Source
Properties
Java Beans
Отдел Кадров
Нанять сотрудника
Бухгалтерия
Нанять сотрудника

Visual Editor. Типовые операции.

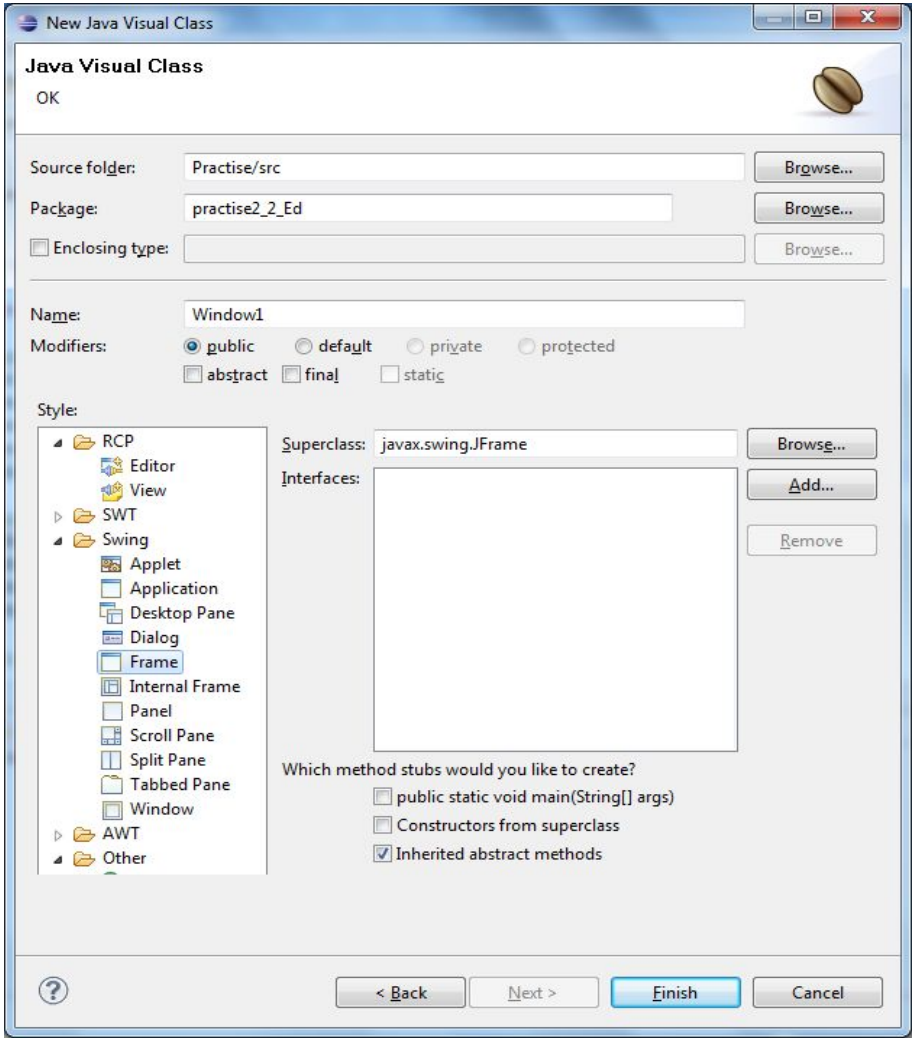
Открыть класс в редакторе

Чтобы открыть существующий класс в визуальном редакторе:

- **Package Explorer** - выберите исходный файл .java
- **Open With > Visual Editor** из выпадающего меню.

Создание нового визуального класса

- **File -> New -> Visual Class** -
- **Name -** имя класса
- **Style –** выбрать родительский класс:
  - JFrame
  - Japplet
  - …..

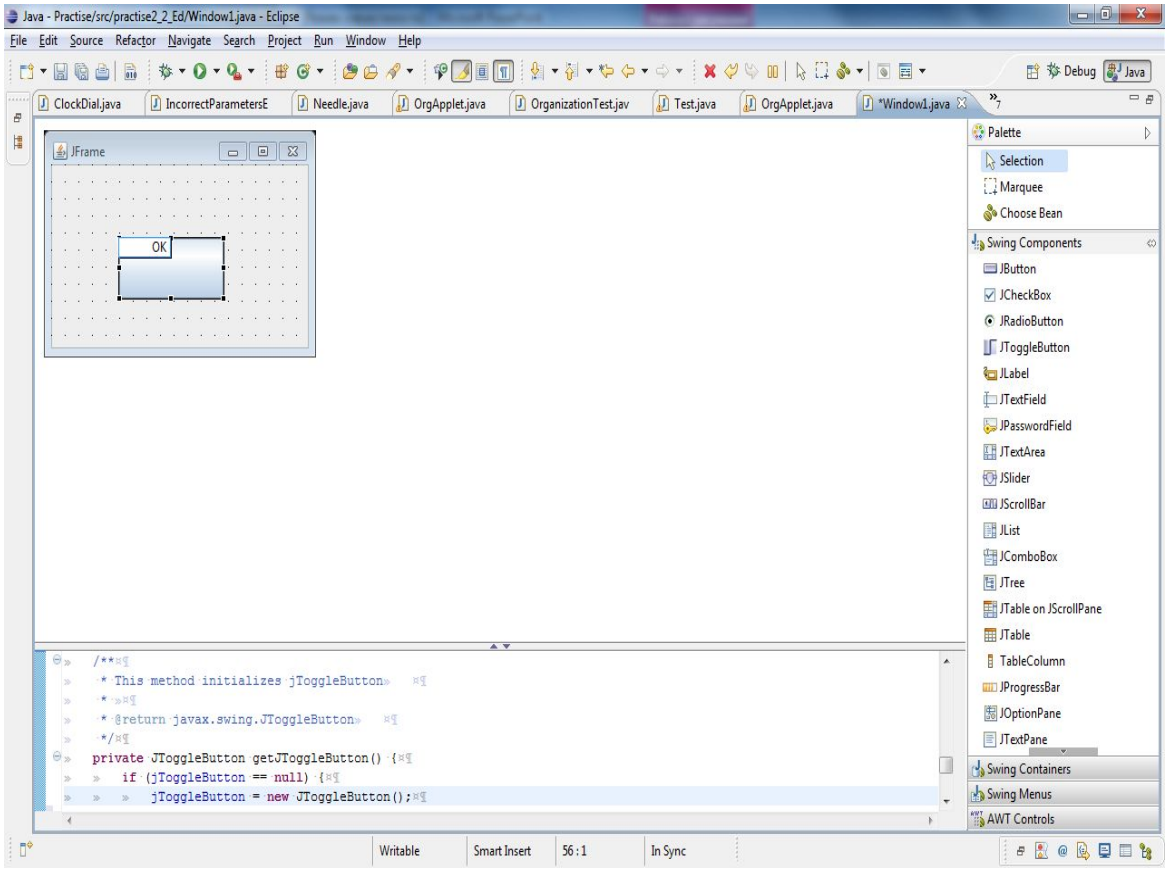

New Java Visual Class
Java Visual Class
OK
Source folder:
Practise/src
Browse...
Package:
practise2_2_Ed
Browse...
Enclosing type:
Browse...
Name:
Window1
Modifiers:
public
default
private
protected
abstract
final
static
Style:
RCP
Editor
View
SWT
Swing
Applet
Application
Desktop Pane
Dialog
Frame
Internal Frame
Panel
Scroll Pane
Split Pane
Tabbed Pane
Window
AWT
Other
Superclass:
javax.swing.JFrame
Browse...
Interfaces:
Add...
Remove
Which method stubs would you like to create?
public static void main(String[] args)
Constructors from superclass
Inherited abstract methods
< Back
Next >
Finish
Cancel

Добавление компонентов и изменение их свойств

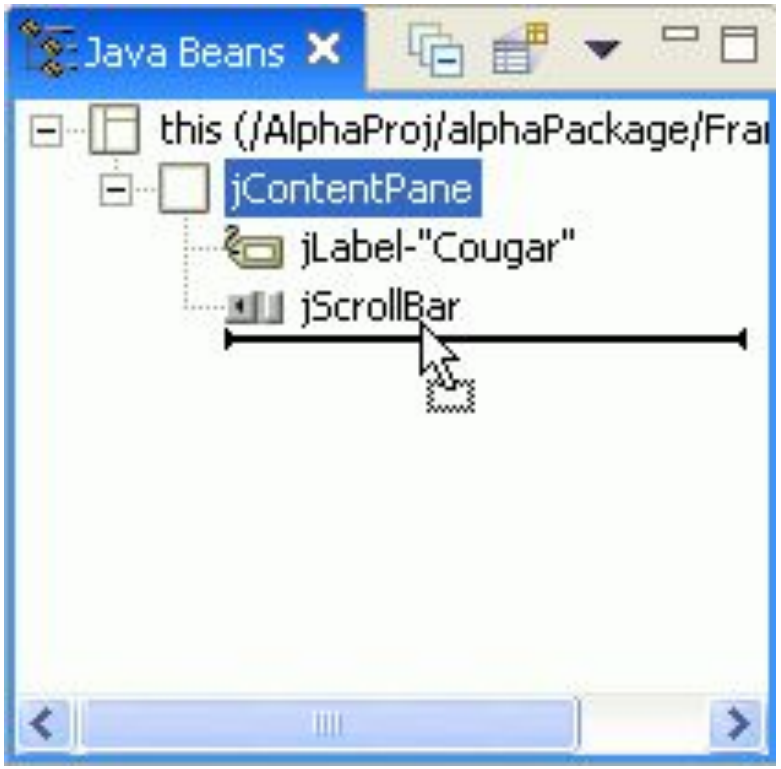
- Добавление компонентов – путем перетаскивания с панели
- Изменение Layout Manager – в контекстном меню панели окна или апплета – **Customize Layout** или **Set Layout**
- Убрать менеджер компонентов - **setLayoutManager(null)** – в коде

  В этом случае можно изменять размещение и размер компонентов вручную

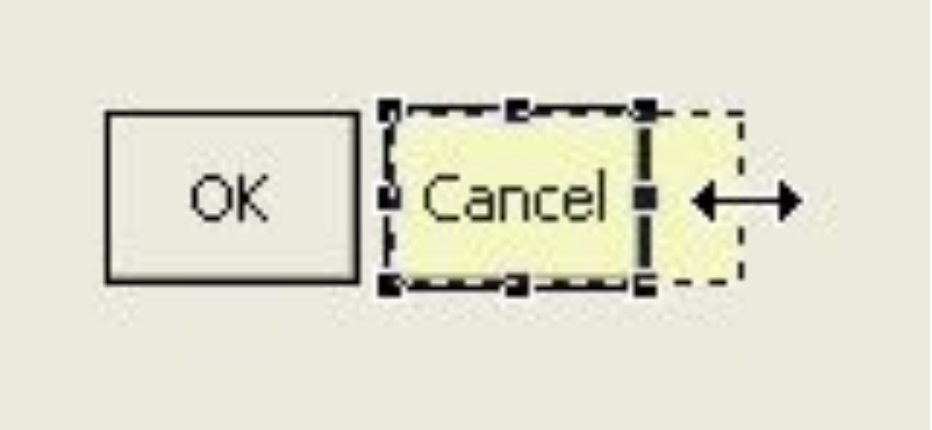
Добавить надпись на компонент



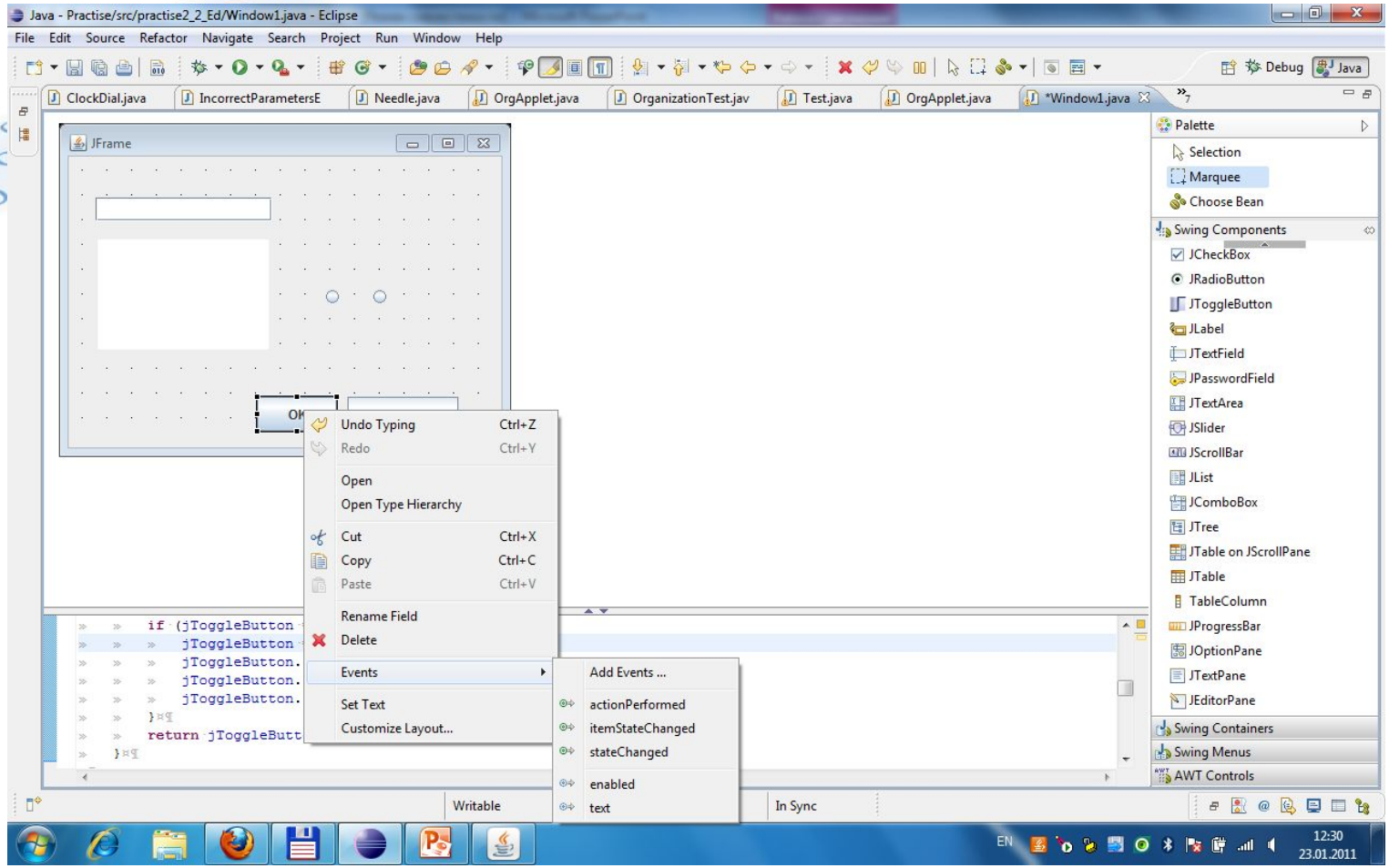
Переупорядочивание компонентов



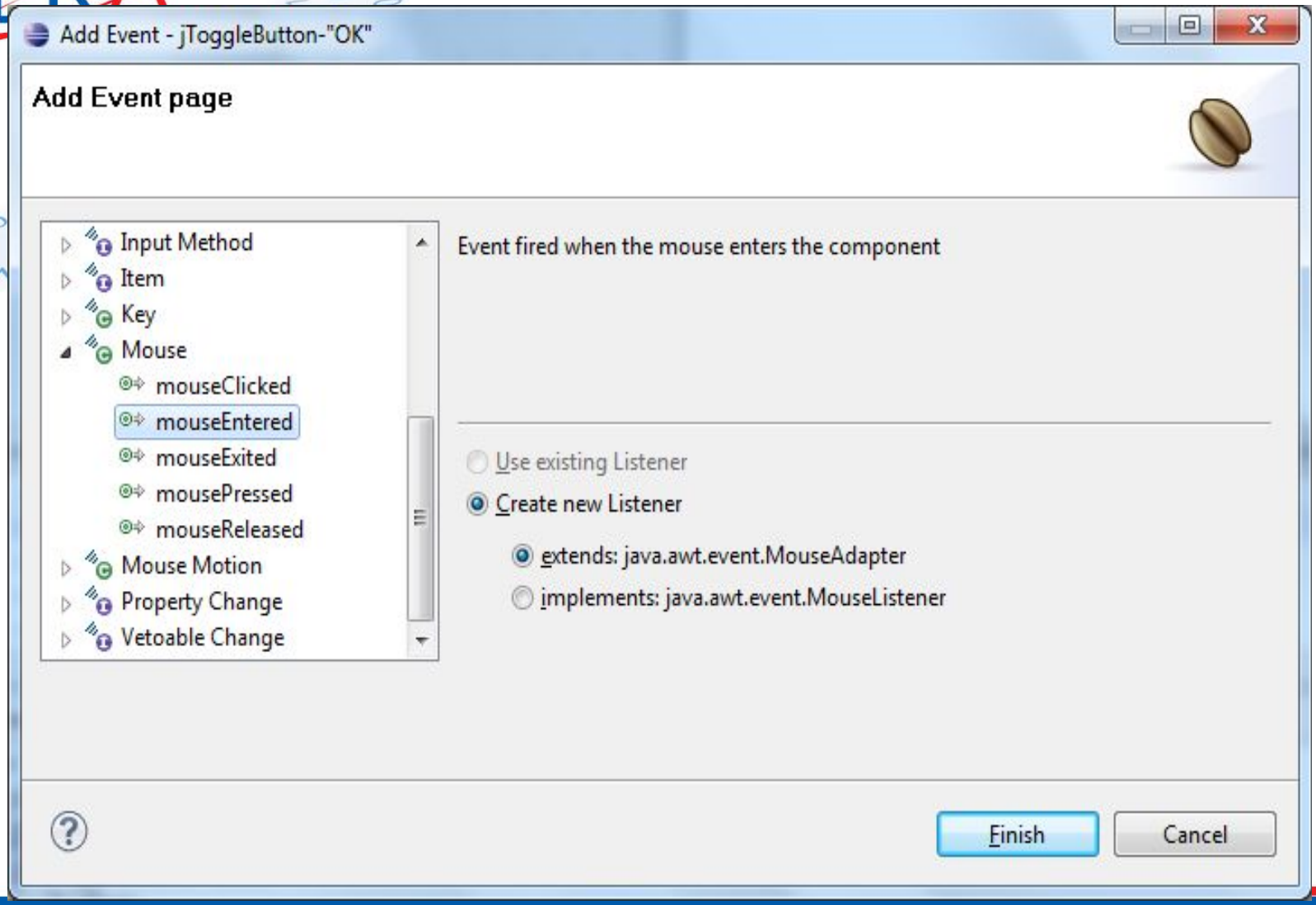
Изменение размеров компонентов



Добавление обработчиков событий



Добавление события



Выравнивание элементов