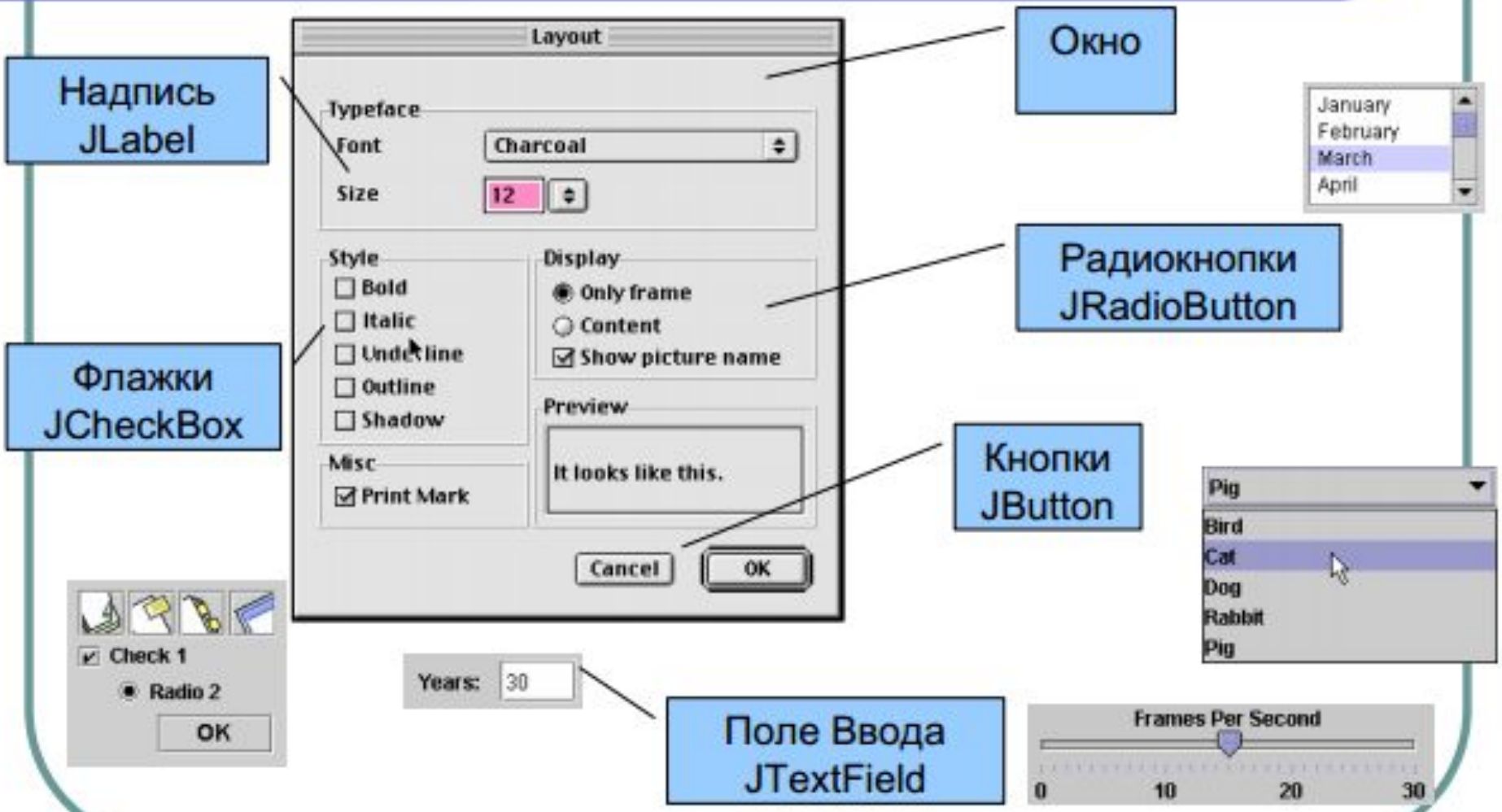
Элементы интерфейса (примеры)



JCheckBox (1) - условие

Выбрано

Не выбрано

Состояние меняется:
при действии пользователя
программно

Конструкторы

JCheckBox()

JCheckBox(метка / иконка / выбрано/не выбрано)

(иконка – изображение формата jpeg, gif, png, представленное как массив байтов)

JCheckBox (2)

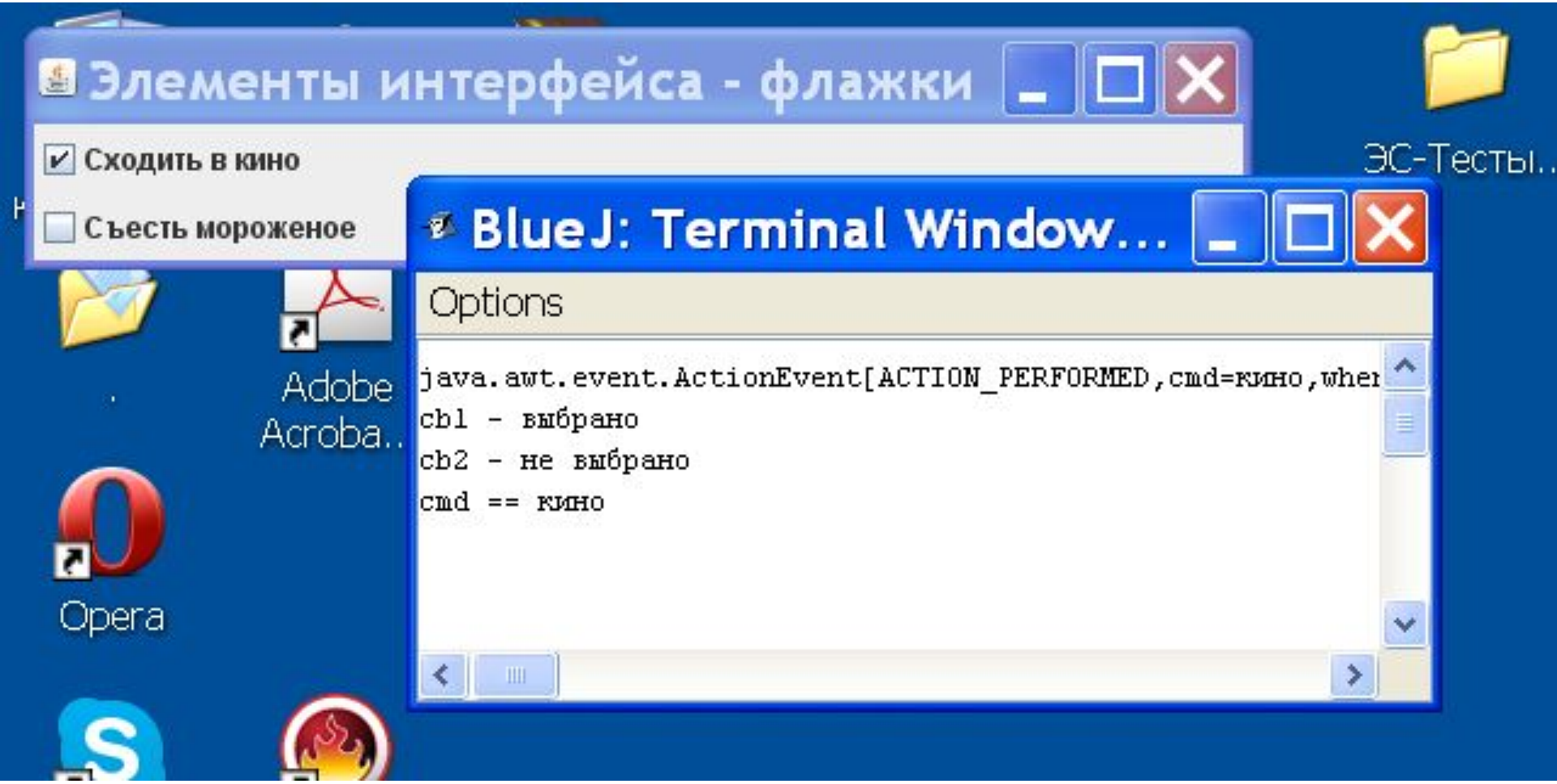
- Проверка состояния – isSelected()
- При изменении состояния – ActionEvent
- Обработка события в actionPerformed (e)
  - как для кнопки – анализ e.getActionCommand (== метке)
  - получением источника события: для cb1

```
Object источник = e.getSource( )
if (источник== cb1) {
boolean newState = ((JCheckBox)cb1).isSelected( );
…}
```

```
1  import java.awt.*; // Проект  - 1 GUI_флажки
2  import javax.swing.*;
3  import java.awt.event.*;
4  public class GUI_CB1 extends JComponent
5                                  implements ActionListener{
6      JCheckBox cb1, cb2;
7      public GUI_CB1(){ // конструктор
8          cb1 = new JCheckBox("Сходить в кино");
9          cb1.setActionCommand("кино");
10         cb2 = new JCheckBox("Съесть мороженое");
11        cb2.setActionCommand("мороженое");
12         cb1.addActionListener(this);
13         cb2.addActionListener(this);
14   }
```

```
15    public void actionPerformed (ActionEvent e){
16      System.out.println(e);
17
18      if(cb1.isSelected())
19            System.out.println("cb1 - выбрано");
20      else  System.out.println("cb1 - не выбрано");
21
22      if(cb2.isSelected())
23            System.out.println("cb2 - выбрано");
24      else  System.out.println("cb2 - не выбрано");
25
26      if("кино".equals(e.getActionCommand()))
27            System.out.println ("cmd == кино");
28      if("мороженое".equals(e.getActionCommand()))
29            System.out.println ("cmd == мороженое");
30      System.out.println();
31    }
32  }
```

```
import java.awt.*;
import javax.swing.*;
public class MyFrame{
  private static void createAndShowGUI(){
     JFrame frame  = new JFrame("Элементы интерфейса – флажки");
     frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
     Container myC = frame.getContentPane();
     GUI_CB1 s = new GUI_CB1(); //сам компонент s – не визуальный,
     myC.setLayout(new GridLayout(2,1)); // размещение таблицей
     myC.add(s.cb1); //добавление кнопок на панель контента
     myC.add(s.cb2); // если просто добавить s, кнопок видно не будет
     frame.setSize(200,100);
     frame.setLocation(10,10);
     frame.setVisible(true);
  }
  public static void main (String[ ] args){
     javax.swing.SwingUtilities.invokeLater(new Runnable() {
        public void run(){createAndShowGUI();}});
  }
}
```


Элементы интерфейса - флажки
Сходить в кино
Съесть мороженое
BlueJ: Terminal Window...
Options
java.awt.event.ActionEvent[ACTION_PERFORMED,cmd=кино,when
cb1 - выбрано
cb2 - не выбрано
cmd == кино
ЭС-Тесты..
Adobe Acroba..
Opera

JRadioButton (1)

- Радиокнопка – разновидность обычной кнопки (команды, подсказки, setEnabled и т.д.)
- Щелчок по кнопке → ActionEvent e
- Кнопки группируются:

```
ButtonGroup g = new ButtonGroup( );
g.add (первая кнопка);
…
g.add (последняя кнопка);
```

- Только одна кнопка в группе м.б. "нажата"

JRadioButton (2)

Типовая последовательность действий:

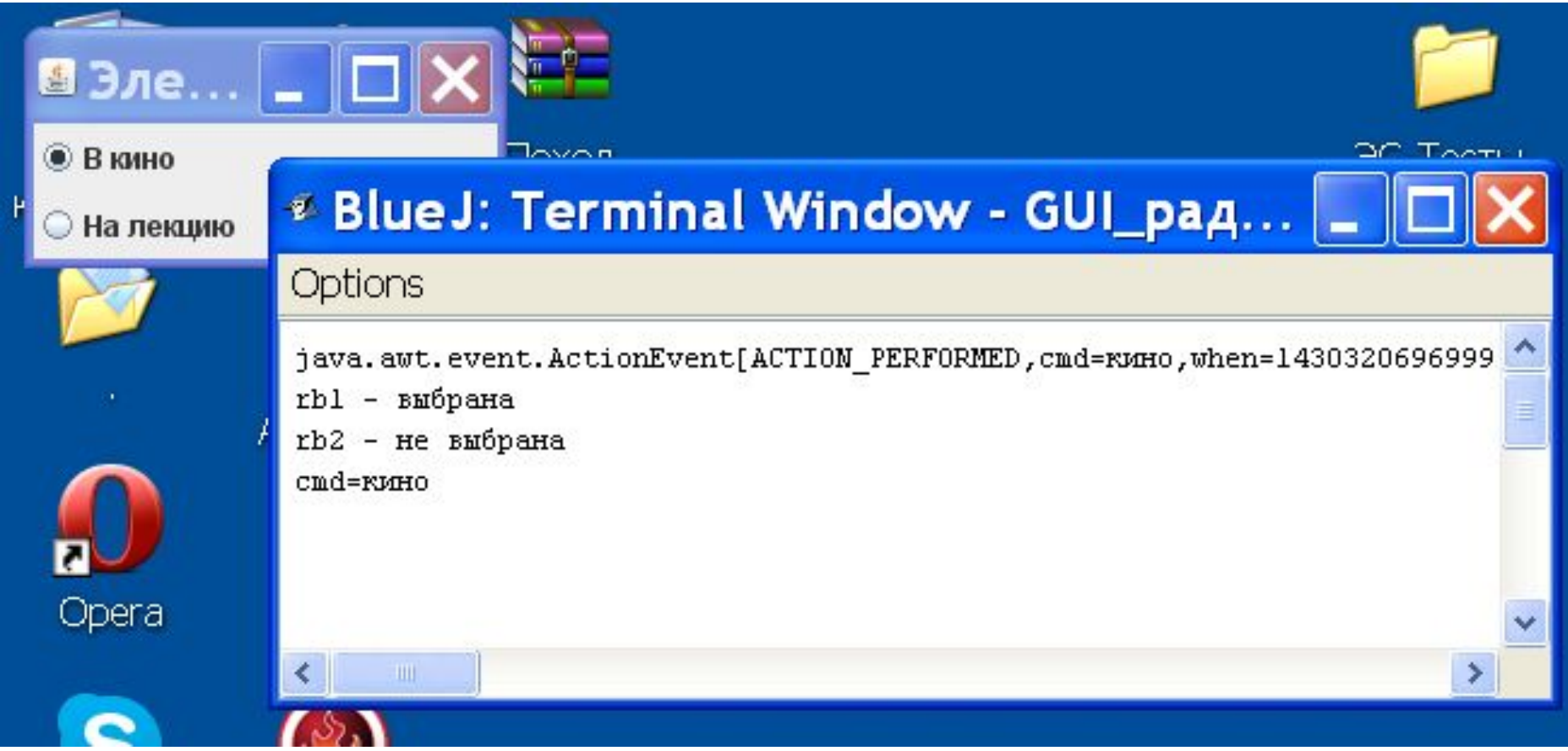
- Создать каждую кнопку
  - назначить команду
  - зарегистрировать слушателя
  - задать начальное состояние (df – выкл.)
  - задать подсказку и мнемонику
- Создать группу и добавить кнопки в группу

```
1   import java.awt.*; //Проект - 2 GUI_радио_кнопки
2   import javax.swing.*;
3   import java.awt.event.*;
4   public class GUI_RB1 extends JComponent
5                                   implements ActionListener{
6   JRadioButton rb1, rb2;
7   public GUI_RB1(){ //конструктор
8       rb1 = new JRadioButton ("В кино");
9       rb1.setActionCommand("кино");
10      rb2 = new JRadioButton ("На лекцию");
11      rb2.setActionCommand("лекция");
12      rb2.setSelected(true);
13      rb1.addActionListener(this);
14      rb2.addActionListener(this);
15      ButtonGroup group = new ButtonGroup();
16      group.add(rb1);     group.add(rb2);
17      //  а что будет, если не создать группу?
18  }
```

Радиокнопки, не включенные в группу, подобны флажкам, т.е. может быть выбрано сразу несколько кнопок

```
19    public void actionPerformed (ActionEvent e){
20        System.out.println(e);
21
22        if (rb1.isSelected())
23            System.out.println("rb1 - выбрана");
24        else  System.out.println("rb1 - не выбрана");
25
26        if (rb2.isSelected())
27            System.out.println("rb2 - выбрана");
28        else  System.out.println("rb2 - не выбрана");
29
30        if ("кино".equals(e.getActionCommand()))
31            System.out.println("cmd = кино");
32        if ("лекция".equals(e.getActionCommand()))
33            System.out.println("cmd = лекция");
34        System.out.println();
35    }
36 }
```

```
import java.awt.*;
import javax.swing.*;
public class MyFrame{
  private static void createAndShowGUI(){
     GUI_RB1 s = new GUI_RB1();
     JFrame frame = new JFrame("Элементы интерфейса-радиокнопки");
     frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
     Container myC = frame.getContentPane();
     myC.setLayout(new GridLayout(2,1));
     myC.add(s.rb1);
     myC.add(s.rb2);
     frame.setSize(200,100);
     frame.setLocation(10,10);
     frame.setVisible(true);
  }
  public static void main (String[ ] args){
     javax.swing.SwingUtilities.invokeLater(new Runnable(){
        public void run(){createAndShowGUI();}});
  }
}
```


Эле...
В кино
На лекцию
BlueJ: Terminal Window - GUI_рад...
Options
java.awt.event.ActionEvent[ACTION_PERFORMED,cmd=кино,when=1430320696999
rb1 - вмбрана
rb2 - не вмбрана
cmd=кино
Opera

Изменим обработчик события (источник события идентифицируем методом e.getSourse()). В этом случае команду на радиокнопку (setActionCommand()) можно не ставить.

```
19    public void actionPerformed (ActionEvent  e){
20        System.out.println(e);
21
22        if (rb1.isSelected())
23            System.out.println("rb1 - выбрана");
24        else  System.out.println("rb1 - не выбрана");
25
26        if (rb2.isSelected())
27            System.out.println("rb2 - выбрана");
28        else  System.out.println("rb2 - не выбрана");
29
30        if (e.getSource() == rb1)
31            System.out.println("cmd = кино");
32        if (e.getSource() == rb2)
33            System.out.println("cmd = лекция");
34        System.out.println();
35    }
36  }
```

Результат – тот же.

Сделаем версию программы с анонимными слушателями событий

```
public class GUI_RB1 extends JComponent { // не включаем интерфейс
JRadioButton rb1, rb2;                    //слушателя
public GUI_RB1(){
rb1 = new JRadioButton ("В кино");
// rb1.setActionCommand("кино"); - можно исключить
rb2 = new JRadioButton ("На лекцию");
// rb2.setActionCommand("лекция"); - можно исключить
rb2.setSelected(true);

rb1.addActionListener (new ActionListener() {
    public void actionPerformed (ActionEvent e){
      System.out.println(e);
      if (rb1.isSelected())
           System.out.println("rb1 - выбрана");
      else  System.out.println("rb1 - не выбрана");
      if (rb2.isSelected())
           System.out.println("rb2 - выбрана");
      else  System.out.println("rb2 - не выбрана");
      System.out.println("cmd = кино");
      System.out.println();
    }});
```

Красным выделена общая часть для обработчиков событий кнопок кнопок rb1 и rb2.

```
rb2.addActionListener (new ActionListener() {
    public void actionPerformed (ActionEvent e){
      System.out.println(e);
      if (rb1.isSelected())
          System.out.println("rb1 - выбрана");
      else  System.out.println("rb1 - не выбрана");
      if (rb2.isSelected())
          System.out.println("rb2 - выбрана");
      else  System.out.println("rb2 - не выбрана");
      System.out.println("cmd = лекция");
      System.out.println();
    }});

ButtonGroup group = new ButtonGroup();
group.add(rb1);
group.add(rb2);
//  а что будет, если не создать группу?
}
}
```

Результат – тот же.

Вывод: когда обработчик содержит большой участок кода, не зависящего от источника события (общего для всех источников), анонимных слушателей использовать невыгодно.

Выпадающий список JComboBox

Выпадающий список — весьма распространенный элемент управления. Он содержит множество вариантов, из которых пользователь может выбрать один и только один, либо (если выпадающий список это позволяет) ввести свой собственный.

Создать выпадающий список можно конструктором по умолчанию JComboBox(), после чего добавлять в него элементы методом addItem(Object item), добавляющим новый элемент в конец списка, или методом insertItemAt(Object item, int index), позволяющим уточнить позицию, в которую требуется вставить элемент.

Однако проще использовать конструктор, в котором сразу указываются все элементы выпадающего списка. Таких конструкторов два: JComboBox(Object[ ] elements) и JComboBox(Vector elements). Работают они одинаково, так что это вопрос удобства разработчика: использовать массив или вектор.

Чаще всего в выпадающий список добавляют строки, однако, как это следует из сигнатур описанных выше методов, он может содержать вообще любые объекты. Любой объект преобразуется к строке методом toString(), именно эта строка и будет представлять его в выпадающем списке.

Метод getItemAt(int index) позволяет обратиться к произвольному элементу (с индексом index в списке).

Метод removeAllItems() удаляет из JComboBox все элементы, а метод removeItem(Object item) — конкретный элемент (при условии, что он содержится в списке).

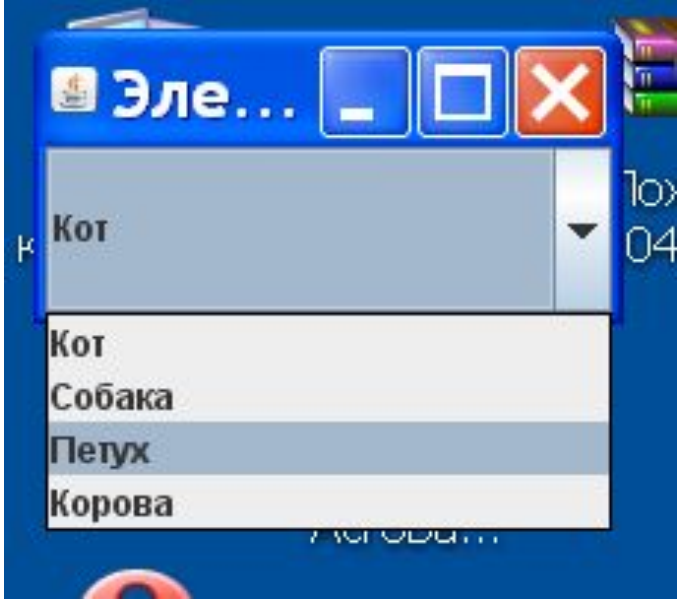
Метод getSelectedIndex() позволяет получить индекс выбранного пользователем элемента (элементы нумеруются начиная с нуля), а метод getSelectedItem() возвращает сам выбранный объект. Сделать конкретный элемент выбранным можно и программно, воспользовавшись методом setSelectedIndex(int index) или setSelectedItem(Object item).

Чтобы пользователь мог ввести свой вариант, который не присутствует в списке, должен быть вызван метод setEditable(boolean editable) с параметром true. Ему соответствует метод isEditable(), который прверяет возмжность редактирования.

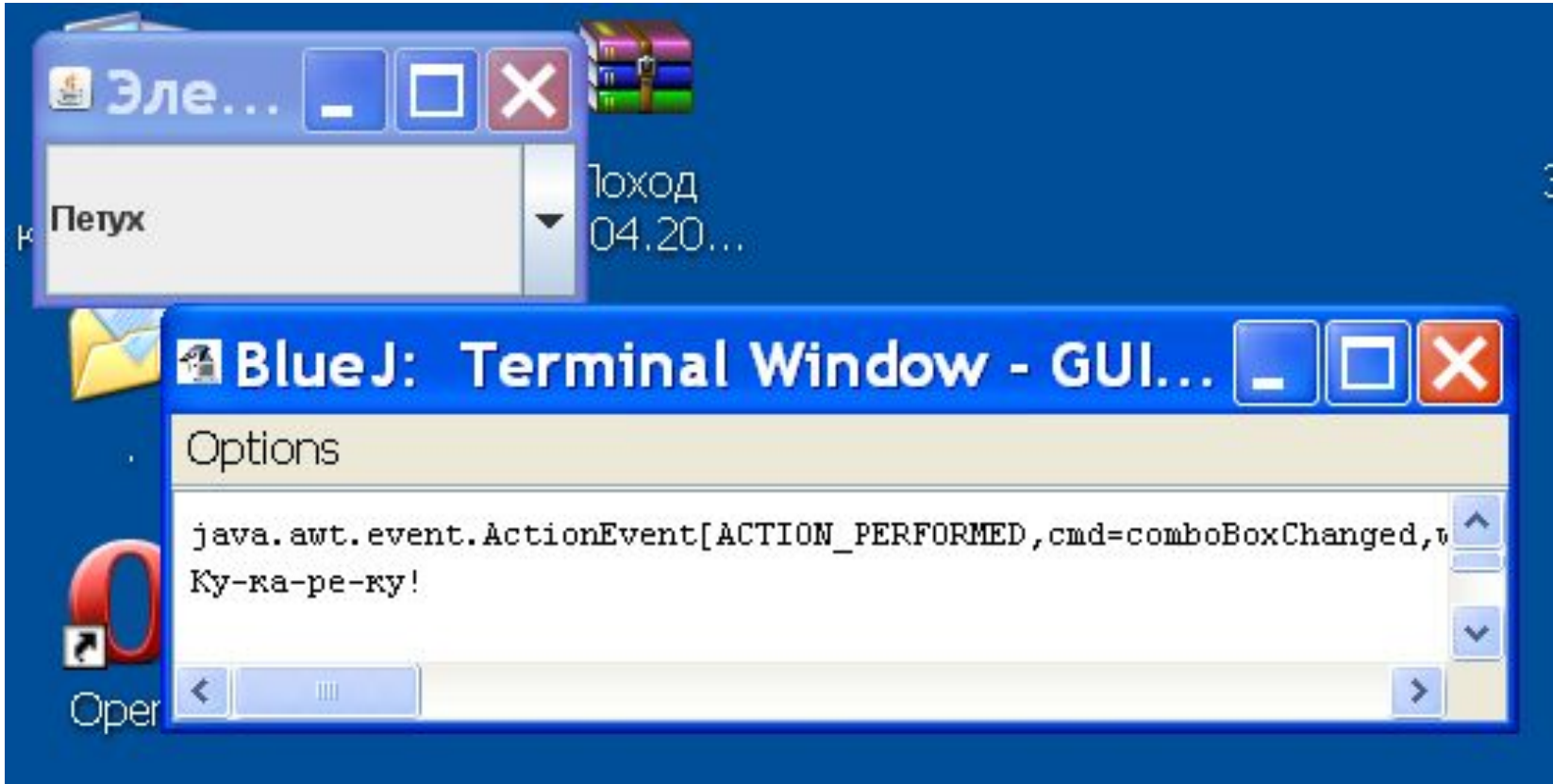
```
import java.awt.*; //Проект - 3 GUI_выпадающий список 1
import javax.swing.*;
import java.awt.event.*;
import javax.swing.event.*;

public class GUI_Combo extends JComponent
                                implements ActionListener {
   JComboBox combo;
   public GUI_Combo() { //конструктор
      String[ ] elements = new String[ ] {"Кот", "Собака",
                                          "Петух", "Корова"};
      combo = new JComboBox(elements);
      combo.setSelectedIndex(0);
      combo.addActionListener(this);
  }
```

```
public void actionPerformed (ActionEvent e){
    System.out.println(e);
    int i = combo.getSelectedIndex(); //получаем индекс
                                      //выбранного элемента
    switch (i) {
        case 0: System.out.println ("Мяу-мяу!"); break;
        case 1: System.out.println ("Гав-гав!"); break;
        case 2: System.out.println ("Ку-ка-ре-ку!"); break;
        case 3: System.out.println ("Му-у-у!"); break;
    }
  }
}
```

```
import java.awt.*;
import javax.swing.*;
public class MyFrame{
  private static void createAndShowGUI(){
     JFrame frame = new JFrame("Элементы интерфейса");
     frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
     Container myC = frame.getContentPane();
     GUI_Combo s = new GUI_Combo();
     myC.add(s.combo);
     frame.setSize(200,100);
     frame.setLocation(10,10);
     frame.setVisible(true);
  }
  public static void main (String[ ] args){
     javax.swing.SwingUtilities.invokeLater(new Runnable(){
        public void run(){createAndShowGUI();}});
  }
}
```



Щелчок мышью



```
import java.awt.*; // Проект - 4 GUI_выпадающий список с панелью 2
import javax.swing.*;
import java.awt.event.*;
import javax.swing.event.*;
```

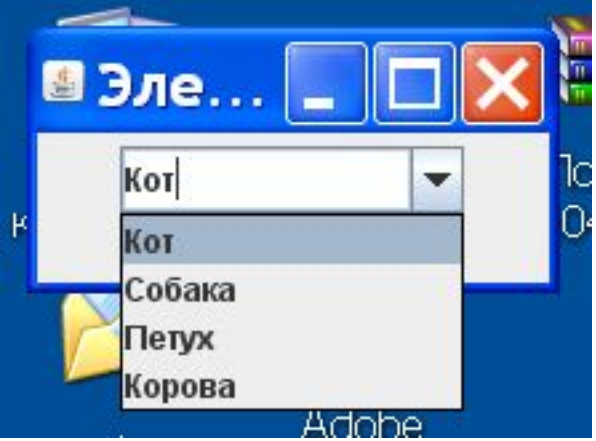
Выпадающий список с панелью

```
public class GUI_Combo extends JPanel
                implements ActionListener {
  JComboBox combo;
  public GUI_Combo() { //конструктор
  String[ ] elements = new String[ ] {"Кот", "Собака",
                                      "Петух", "Корова"};
  combo = new JComboBox(elements);
  combo.setSelectedIndex(0);
  combo.setEditable(true); //разрешить ввод и
                           //редактирование значения
                           //в текстовой строке списка
  combo.addActionListener(this);
  add(combo); // список добавлен к панели }
```

```
  public void actionPerformed (ActionEvent e){
     System.out.println(e);
     int i = combo.getSelectedIndex();
     switch (i){
        case 0: System.out.println ("Мяу-мяу!"); break;
        case 1: System.out.println ("Гав-гав!"); break;
        case 2: System.out.println ("Ку-ка-ре-ку!"); break;
        case 3: System.out.println ("Му-у-у!"); break;
        default: System.out.println ("Это что-то другое");
     }
  }
}
```

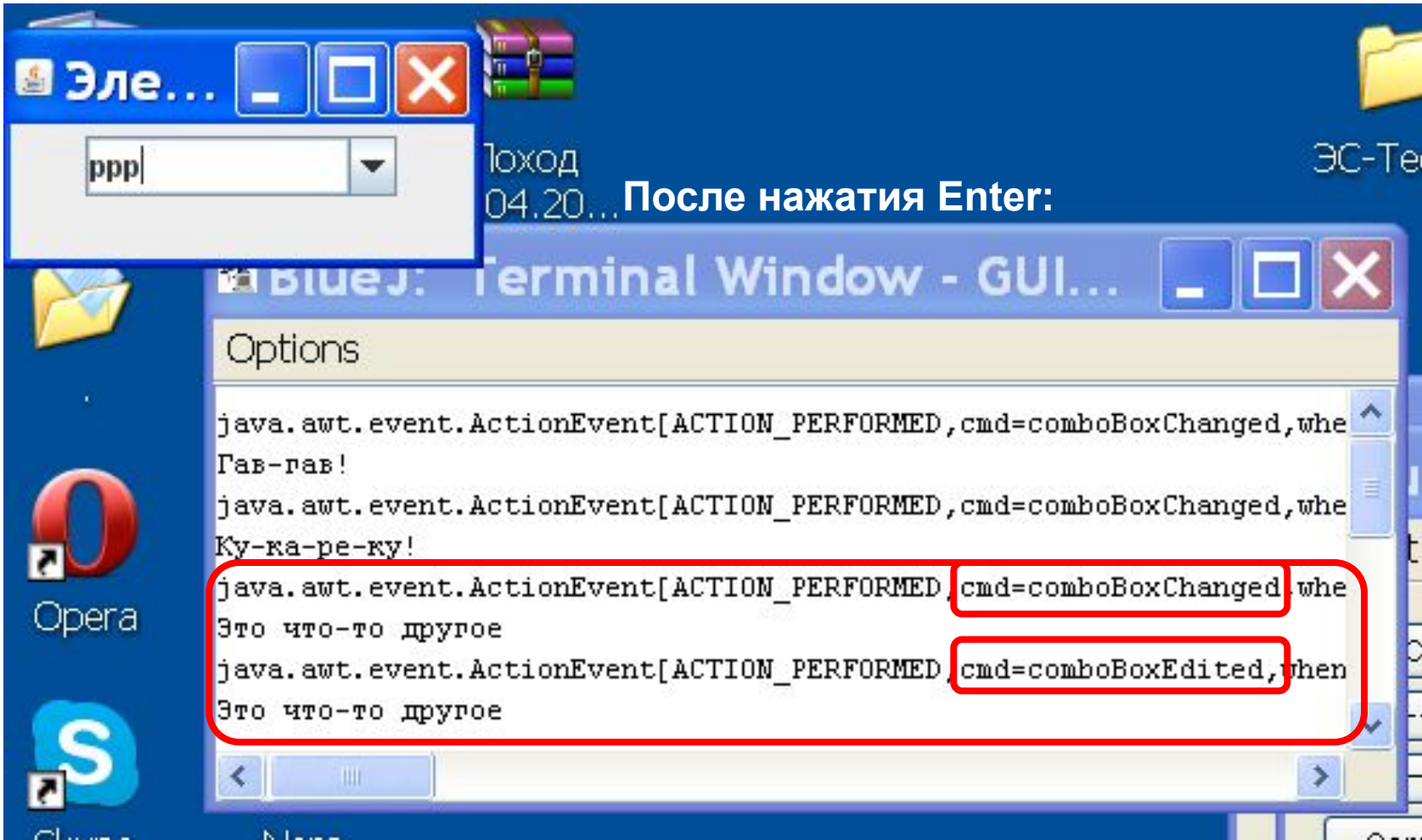
Здесь ничего не изменилось

```
import java.awt.*;
import javax.swing.*;
public class MyFrame{
  private static void createAndShowGUI(){
     JFrame frame = new JFrame("Элементы интерфейса");
     frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
     Container myC = frame.getContentPane();
     GUI_Combo p = new GUI_Combo(); // это панель – видимый
                  //компонент, на который добавлены (add) кнопки
     myC.add(p); //добавили панель p к панели контента фрейма
     frame.setSize(200,100);
     frame.setLocation(10,10);
     frame.setVisible(true);
  }
  public static void main (String[ ] args){
     javax.swing.SwingUtilities.invokeLater(new Runnable(){
        public void run(){createAndShowGUI();}}); }}
```



Сначала поочередно выбрали из списка собаку и петуха, а потом набрали в текстовой строке списка “ppp”.

Почему при вводе строки “ppp” (нажатии «Enter») сообщение «Это что-то другое» вывелось 2 раза? Проанализируйте cmd возникающих событий (в окне терминала).

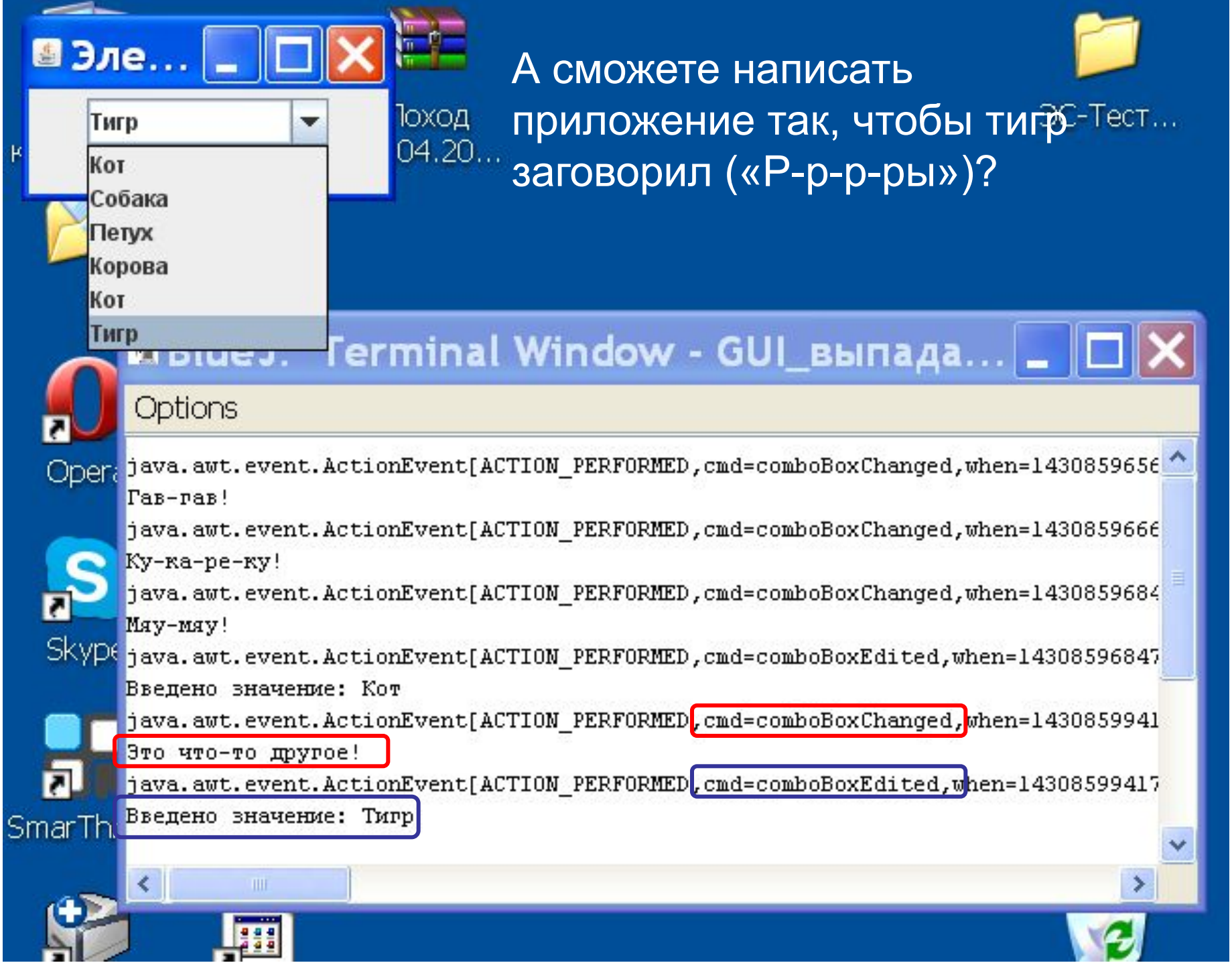


Можно реагировать на каждую разновидность события ActionEvent.

Можно обращаться к отдельным элементам JComboBox, например, к его текстовому редактору.

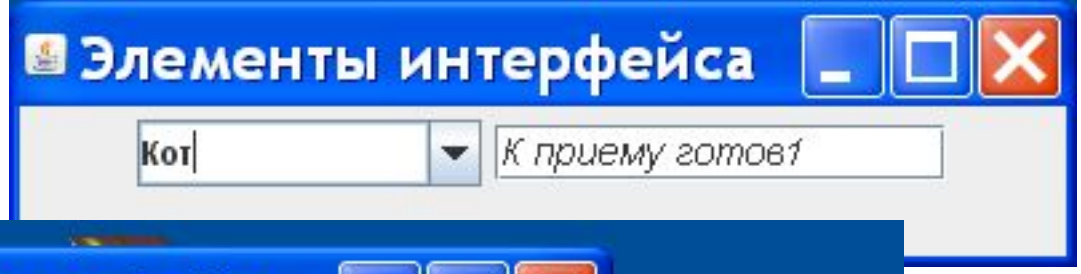
```
// Проект - 5 GUI_выпадающий список с панелью
// и раздельной обработкой событий 3
import java.awt.*; import javax.swing.*; import java.awt.event.*;
import javax.swing.event.*;
public class GUI_Combo extends JPanel
                                    implements ActionListener {
   JComboBox combo;
   JTextField editor;
   public GUI_Combo() { //конструктор
      String[ ] elements = new String[ ] {"Кот", "Собака",
                           "Петух", "Корова"};
      combo = new JComboBox(elements);
      combo.setSelectedIndex(0);
      combo.setEditable(true);
      editor = (JTextField) combo.getEditor().getEditorComponent();
      editor.setEditable(true);
      combo.addActionListener(this);
      add (combo); //добавили на панель}
```

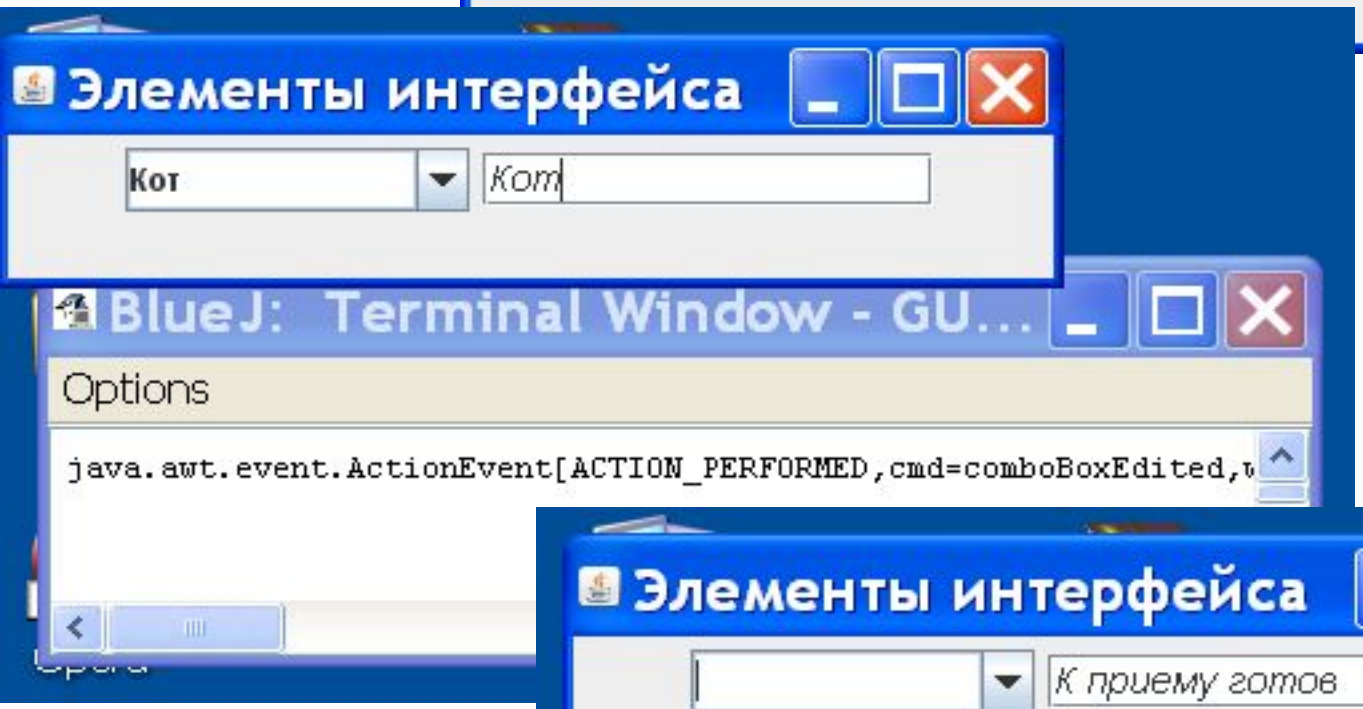
```
public void actionPerformed (ActionEvent e){
   System.out.println(e);
   if("comboBoxChanged".equals(e.getActionCommand())){
      int i = combo.getSelectedIndex();
      switch (i){
         case 0: System.out.println ("Мяу-мяу!"); break;
         case 1: System.out.println ("Гав-гав!"); break;
         case 2: System.out.println ("Ку-ка-ре-ку!"); break;
         case 3: System.out.println ("Му-у-у!"); break;
         default: System.out.println ("Это что-то другое!");
      }
  }
  else {System.out.println (  //cmd=comboBoxEdited
      "Введено значение: " + editor.getText());
      combo.addItem(editor.getText()); //добавлен пункт
                                       //в список
  }
}} //Класс MyFrame – без изменений.
```


Эле...
Тигр
Кот
Собака
Петух
Корова
Кот
Тигр
Поход 04.20...
ЭС-Тест...
А сможете написать приложение так, чтобы тигр заговорил («Р-р-р-ры»)?
Terminal Window - GUI_выпада...
Options
java.awt.event.ActionEvent[ACTION_PERFORMED,cmd=comboBoxChanged,when=1430859656
Гав-гав!
java.awt.event.ActionEvent[ACTION_PERFORMED,cmd=comboBoxChanged,when=1430859666
Ку-ка-ре-ку!
java.awt.event.ActionEvent[ACTION_PERFORMED,cmd=comboBoxChanged,when=1430859684
Мяу-мяу!
java.awt.event.ActionEvent[ACTION_PERFORMED,cmd=comboBoxEdited,when=1430859684
Введено значение: Кот
java.awt.event.ActionEvent[ACTION_PERFORMED,cmd=comboBoxChanged,when=1430859941
Это что-то другое!
java.awt.event.ActionEvent[ACTION_PERFORMED,cmd=comboBoxEdited,when=1430859941
Введено значение: Тигр

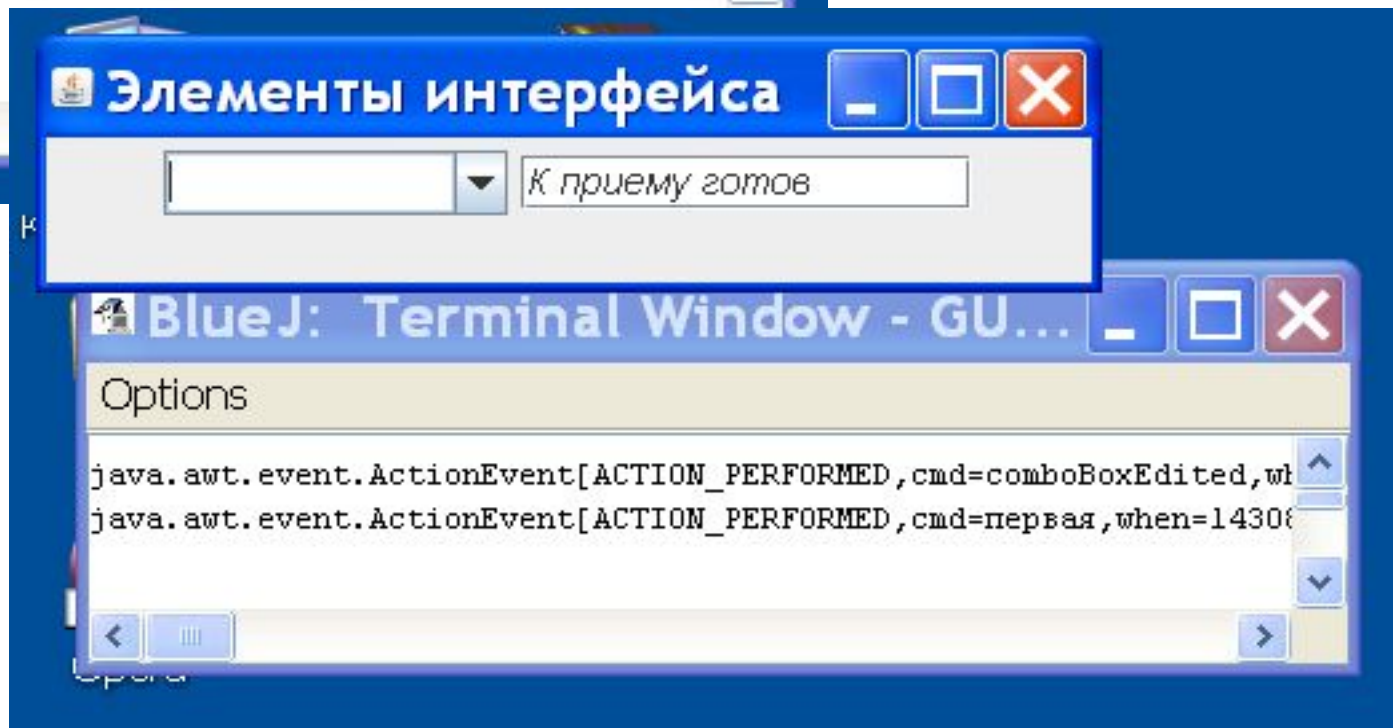
В следующем примере строка из текстового редактора элемента JComboBox передается в текстовое поле JTextField.

<Enter>





<Enter>



Можно вручную ввести в editor строку или выбрать строку из списка.

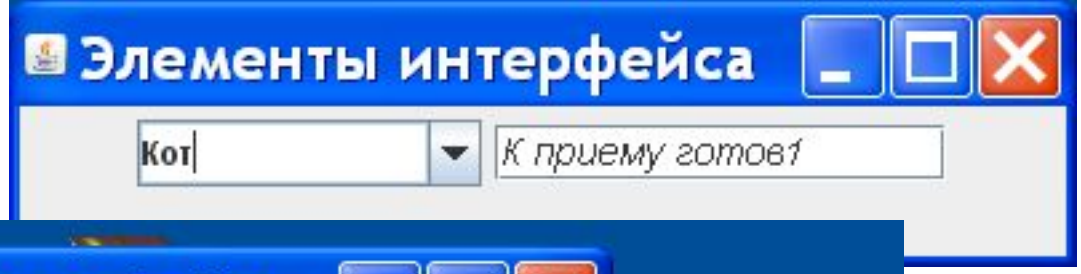
```
// Проект - 6 GUI_выпадающий список и текстовое поле 4
import java.awt.*; import javax.swing.*;
import java.awt.event.*; import javax.swing.event.*;
public class GUI_Combo extends JPanel
                                    implements ActionListener {
   JComboBox combo;
   JTextField editor;  JTextField tf1;
   public GUI_Combo() { //конструктор
   String[ ] elements = new String[ ] {"Кот", "Собака",
                                          "Петух", "Корова"};
   combo = new JComboBox(elements);
   combo.setSelectedIndex(0); combo.setEditable(true);
   editor = (JTextField) combo.getEditor().getEditorComponent();
   editor.setEditable(true); editor.requestFocus();
   combo.addActionListener(this); add (combo);
   tf1 = new JTextField ("К приему готов1", 15);
   tf1.setFont(new Font ("Arial", Font.ITALIC,14));
   tf1.addActionListener(this); tf1.setActionCommand("первая");
   tf1.setEditable(true); add(tf1);
}
```

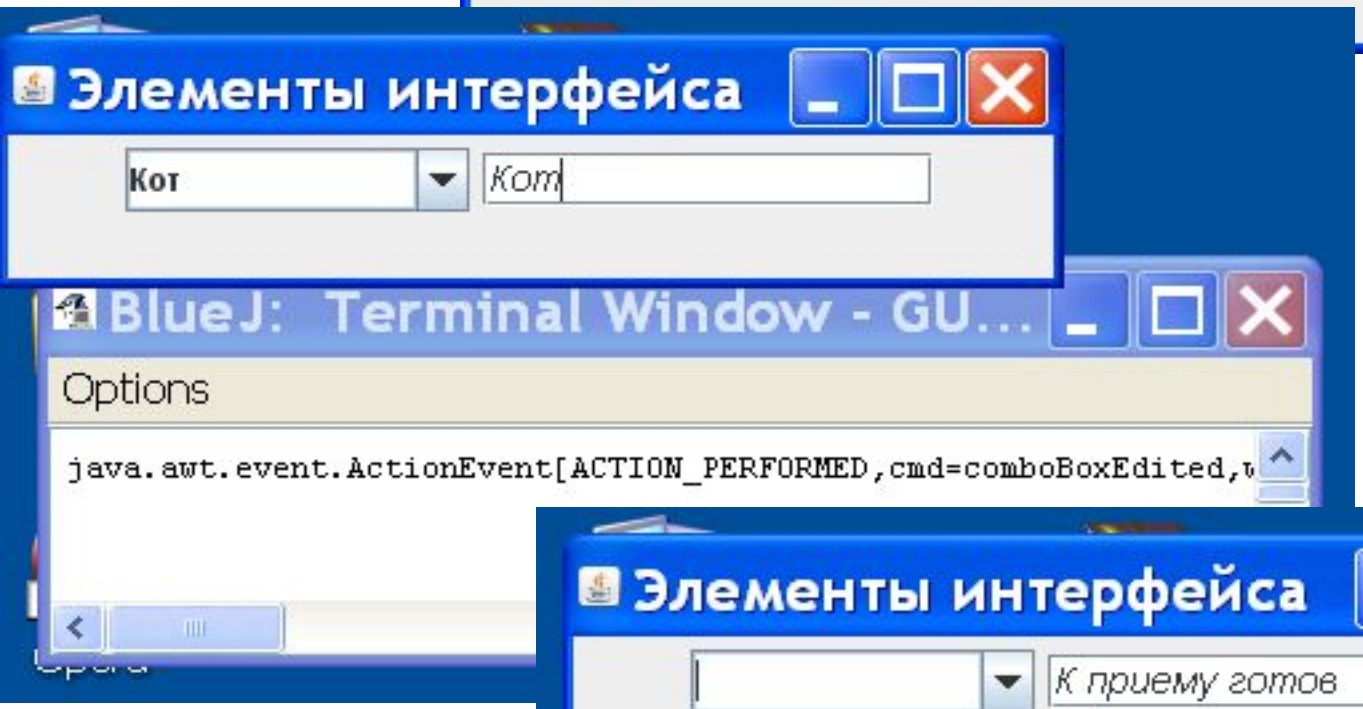
```
public void actionPerformed (ActionEvent e){
    //обработчик событий
    System.out.println(e);
    if("comboBoxEdited".equals(e.getActionCommand())){
      tf1.setText(editor.getText());
      tf1.requestFocus();
    }
    if ("первая".equals(e.getActionCommand())){
      tf1.setText("К приему готов");
      editor.requestFocus();
      editor.setText("");
   }
  }
}
```

Команда по умолчанию

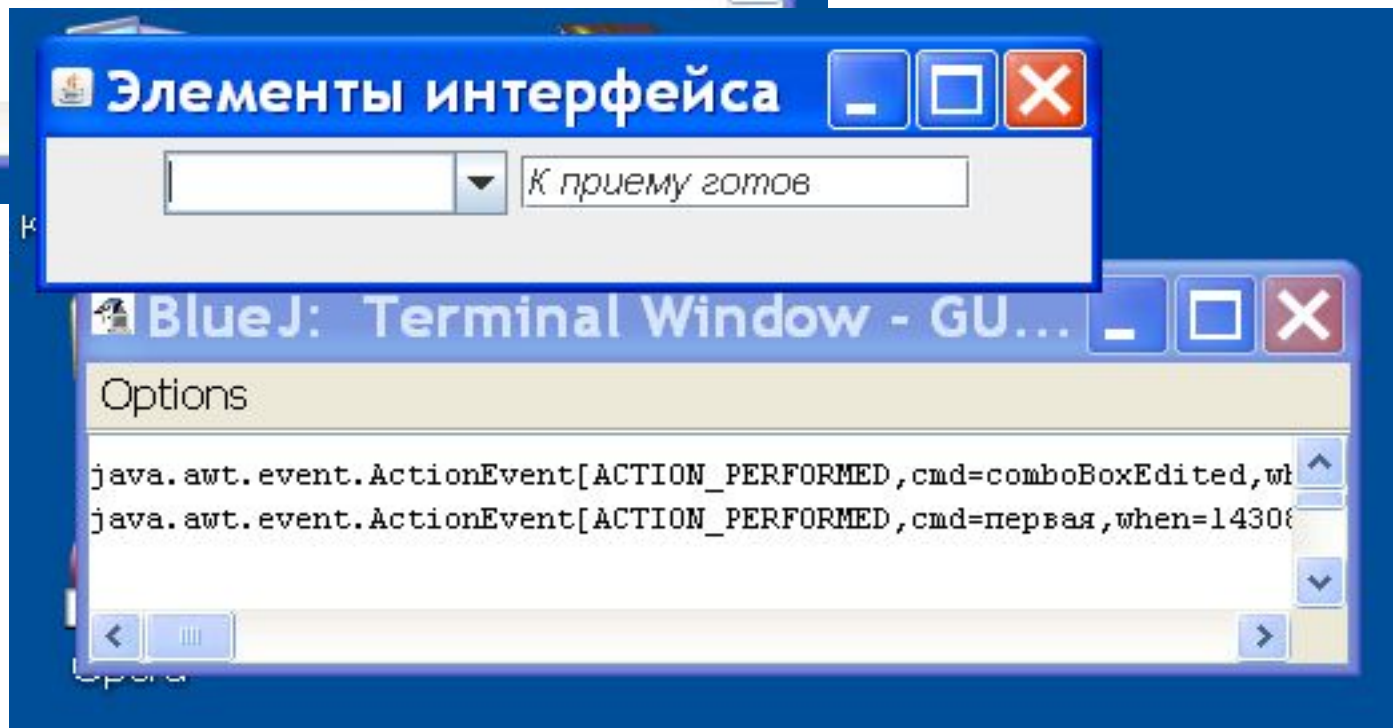
// Класс MyFrame – без изменений.

<Enter>





<Enter>



Можно вручную ввести в editor строку или выбрать строку из списка.

Список JList

Список JList — это один из сложных компонентов, для эффективной работы с которыми необходимо понимание основ библиотеки Swing, в частности, концепции «Модель-Вид». Компоненты JTree (дерево) и JTable (таблица) еще сложнее и будут рассмотрены позднее. Некоторая часть возможностей JList может быть использована без углубления в детали.

Список содержит группу элементов, аналогично выпадающему списку JComboBox, но обладает двумя отличительными особенностями. Во-первых, на экране видны одновременно несколько элементов списка. Во-вторых, пользователь может выбрать в списке не один элемент, а несколько (если установлен соответствующий режим выделения).

Создать список можно с помощью конструктора, работающего на основе массива Object[ ] или вектора Vector (аналогично JComboBox). Метод setVisibleRowCount(int count) устанавливает количество видимых элементов списка. Остальные элементы будут уходить за его пределы или прокручиваться, если поместить список в JScrollPane (что рекомендуется).

По умолчанию пользователь может выбрать в списке любое число элементов, держа нажатой клавишу Ctrl. Это можно изменить, вызвав метод setSelectionMode(int mode), где параметр задается одной из констант класса ListSelectionModel:

SINGLE_SELECTION — может быть выделен только один элемент,

SINGLE_INTERVAL_SELECTION — может быть выделено несколько элементов, но составляющих непрерывный интервал,

MULTIPLE_INTERVAL_SELECTION — может быть выделено произвольное количество смежных и несмежных элементов.

Выделенный элемент списка (если он один) можно получить методом getSelectedValue(). Если таких несколько, метод вернет первый из них. Метод getSelectedValues() возвращает все выделенные элементы списка в виде массива Object [ ]. Аналогично работают методы getSelectedIndex() и getSelectedIndices(), только возвращают они не сами выделенные элементы, а их индексы. Всем этим методам соответствуют методы set, так что выделить элементы списка можно и программно.

JList -- Конструктор

JList(Object[ ] listData)

- Методы
  - addListSelectionListener
  - clearSelection()
  - getFirstVisibleIndex()
  - getLastVisibleIndex()
  - getSelectedIndex()
  - getSelectedValue()

Номер предпоследнего выбранного элемента списка

Номер последнего выбранного элемента списка

ListSelectionEvent -- событие

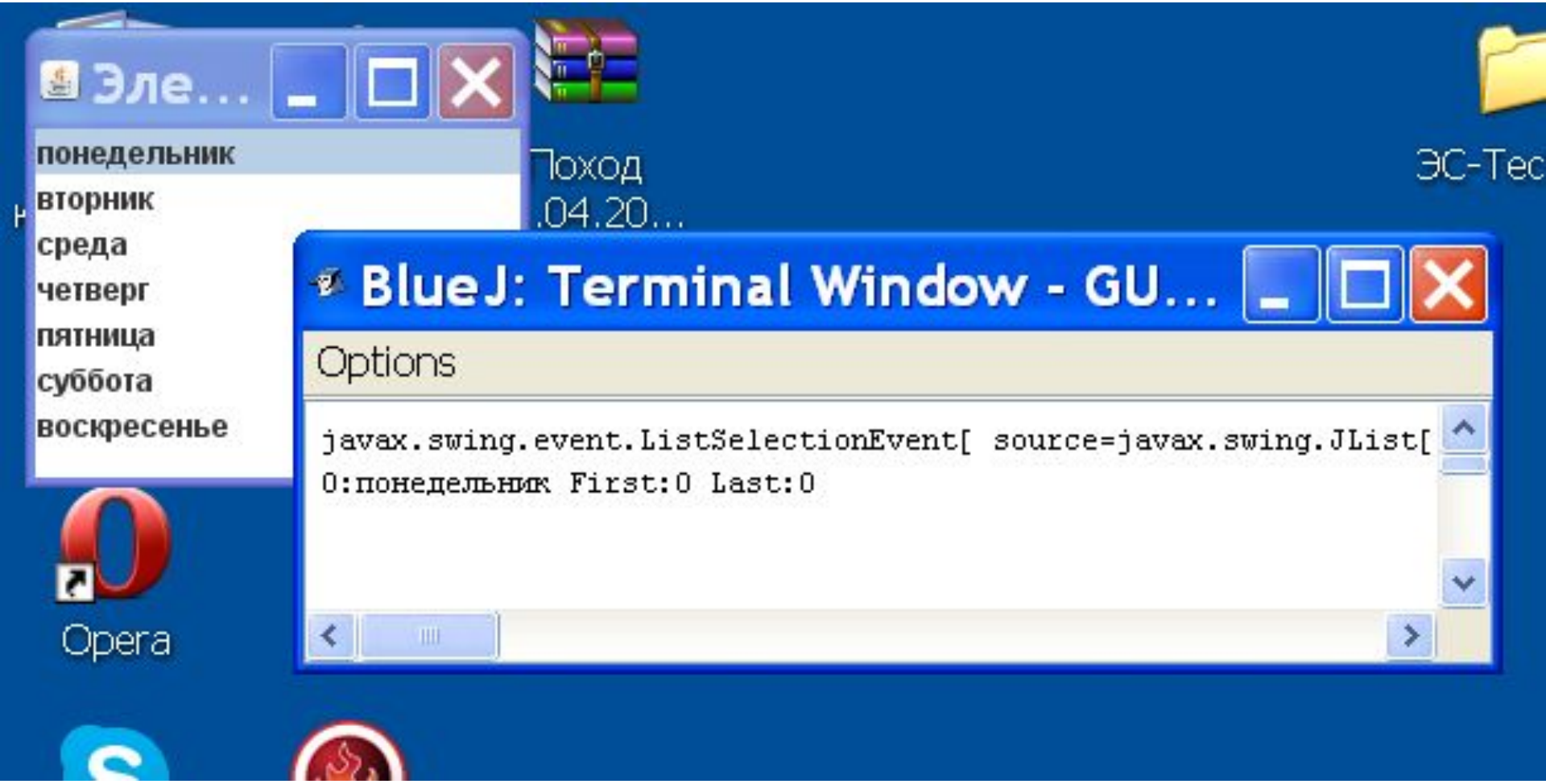
Событие вызывает щелчок мышки или перемещение с клавиатуры (стрелка вверх, стрелка вниз)

- фильтрация - при event.getValueIsAdjusting( ) == false

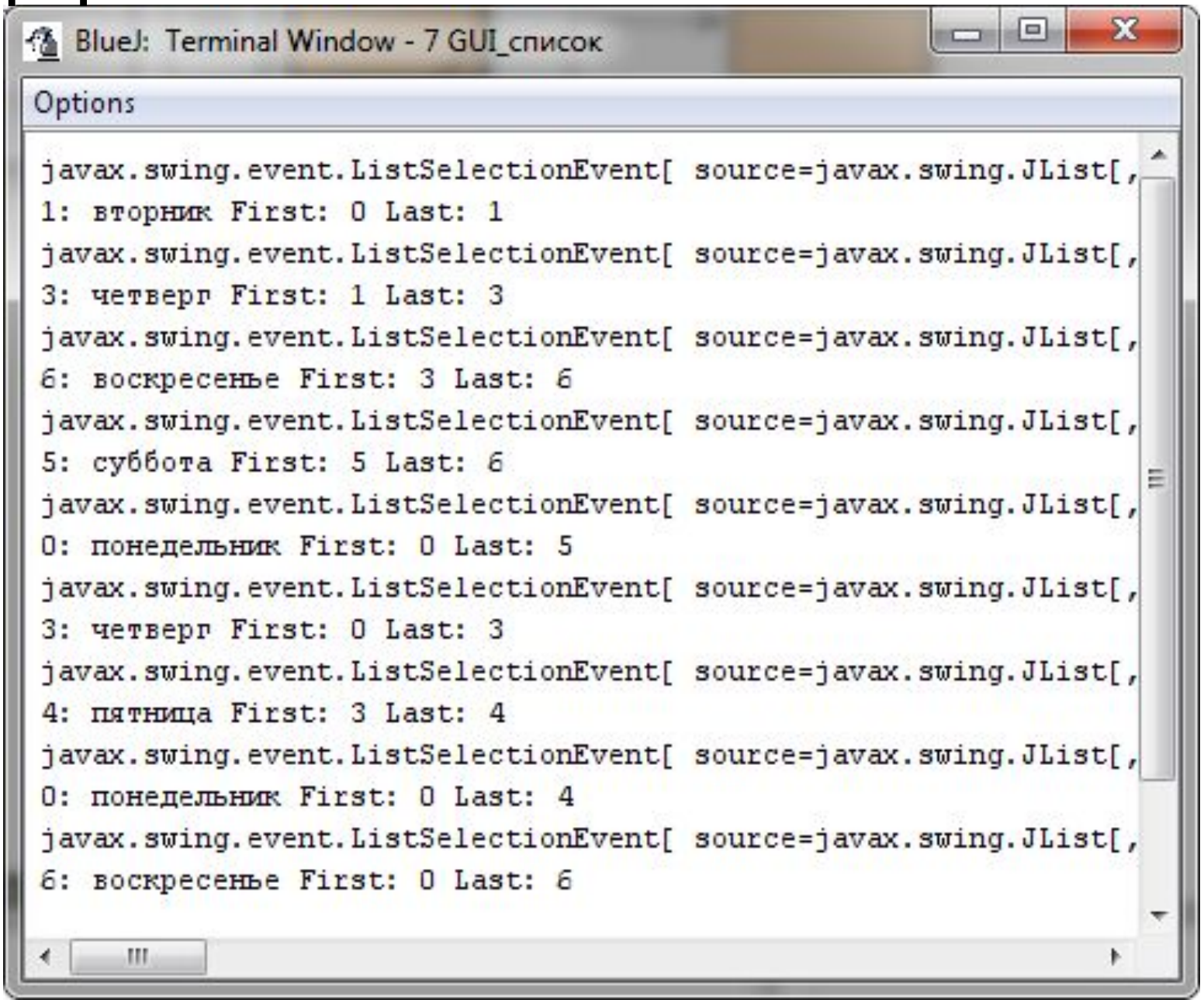
Метод getValueIsAdjusting () возвращает true если событие вызывается по причине выбора нескольких пунктов. Чаще всего нам интересен случай, когда этот метод возвращает false, т.к. нам, как правило, не надо отслеживать многострочные выделения внутри нашего списка.

```
public class GUI_List1 extends JComponent
                                implements ListSelectionListener {
   JList  aL1; //Проект - 7 GUI_список
   public GUI_List1(){
      String дни [ ] = {"понедельник", "вторник","среда",
                        "четверг", "пятница", "суббота",
                        "воскресенье"};
      aL1 = new JList (дни);
      aL1.addListSelectionListener(this);
   }
   public void valueChanged (ListSelectionEvent  e) {
       // обработчик события
       if ( !e.getValueIsAdjusting() ){ //выбран один пункт
          System.out.println(e);
          System.out.printf("%d: %s First: %d Last: %d\n",
          aL1.getSelectedIndex(), aL1.getSelectedValue(),
          e.getFirstIndex(), e.getLastIndex());}
   }}
```

```
import java.awt.*;
import javax.swing.*;
public class MyFrame{
  private static void createAndShowGUI(){
     JFrame frame = new JFrame("Элементы интерфейса");
     frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
     Container myC = frame.getContentPane();
     GUI_List1 s = new GUI_List1();
     myC.add(s.aL1);
     frame.setSize(200,100);
     frame.setLocation(10,10);
     frame.setVisible(true);
  }
  public static void main (String[ ] args){
     javax.swing.SwingUtilities.invokeLater(new Runnable(){
        public void run() {createAndShowGUI();}});
  }
}
```


Эле...
понедельник
вторник
среда
четверг
пятница
суббота
воскресенье
Поход
.04.20...
ЭС-Тес
BlueJ: Terminal Window - GU...
Options
javax.swing.event.ListSelectionEvent[source=javax.swing.JList[
0:понедельник First:0 Last:0
Opera

Исследуйте, как меняются значения First и Last при выборе различных элементов списка.



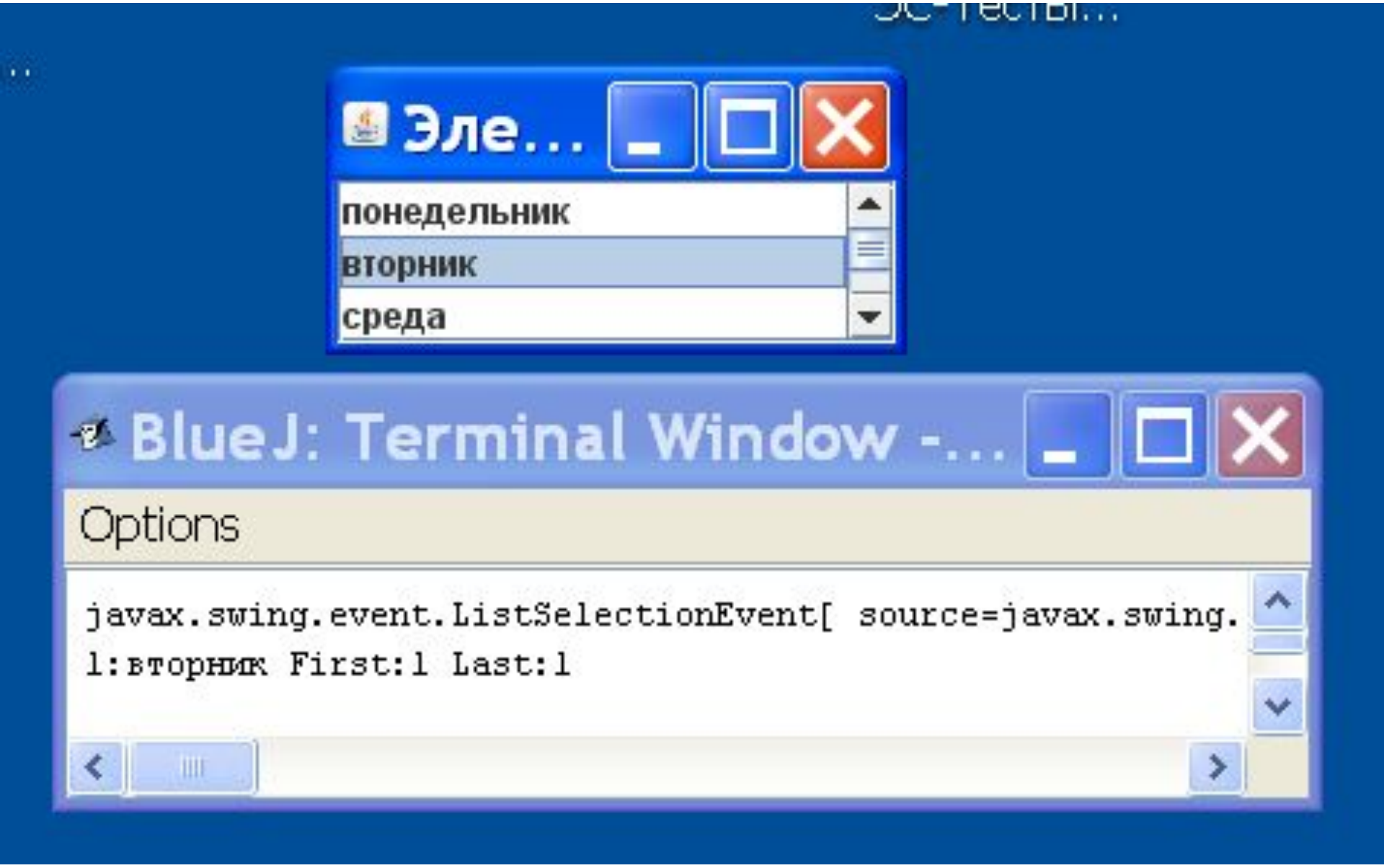
Создадим список с прокруткой (скроллингом).

Панель скроллинга – самостоятельный элемент.

В конструктор скроллинга передается список, для которого создается скроллинг.

Панель скроллинга помещается на панель контента.

```
import java.awt.*; //Проект - 8 GUI_список_прокрутка
import javax.swing.*;
public class MyFrame{
  private static void createAndShowGUI(){
     JFrame frame = new JFrame("Элементы интерфейса");
     frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
     Container myC = frame.getContentPane();
     GUI_List1 s = new GUI_List1();
     // Создаем панель прокрутки для списка
     JScrollPane  scrollPane = new JScrollPane(s.aL1);
     myC.add(scrollPane); //добавляем панель прокрутки со списком
     frame.setSize(200,100);
     frame.setLocation(10,10);
     frame.setVisible(true);
  }
  public static void main (String[ ] args){
     javax.swing.SwingUtilities.invokeLater(new Runnable(){
        public void run() {createAndShowGUI();}});
  }}
```


Эле...
понедельник
вторник
среда
BlueJ: Terminal Window -...
Options
javax.swing.event.ListSelectionEvent[source=javax.swing.
l:вторник First:1 Last:1

Движок – JSlider (1) (ползунок)

- Элемент, который позволяет графически выбрать значение из данного интервала
- Конструкторы (все параметры - int)
  - **JSlider**()
  - **JSlider**(min, max)
  - **JSlider**(min, max, value)
  - **JSlider**(orientation, min, max, value)

Движок – JSlider (2) -- Методы

- **addChangeListener**(ChangeListener l)
- **getMaximum**()
- **getMinimum**()
- **getValue**()
- setMajorTickSpacing(ИНТЕРВАЛ);
  setMinorTickSpacing(интервал);
  setPaintTicks(true/false);
  setPaintLabels(true/false);
- событие – getSource()

```
import java.awt.*; //Проект - 9 GUI_ползунок
import javax.swing.*;
import java.awt.event.*;
import javax.swing.event.*;

public class GUI_Slider extends JPanel
                implements ChangeListener {
   JSlider s1,s2,s3;
   public GUI_Slider() { //конструктор
      s1 = new JSlider (JSlider.HORIZONTAL, -5, 5, 0);
      s1.setMajorTickSpacing(10);//цена большого деления
      s1.setMinorTickSpacing(2); //цена малого деления
      s1.setPaintTicks(true); // отображать деления
      s1.setPaintLabels(true); // отображать метки
```

```
s2 = new JSlider (JSlider.HORIZONTAL,10, 50, 20);
s2.setMajorTickSpacing(10);
s2.setMinorTickSpacing(1);
s2.setPaintTicks(true);
s2.setPaintLabels(true);

s3 = new JSlider (JSlider.VERTICAL, 0, 100, 50);
s3.setMajorTickSpacing(10);
s3.setMinorTickSpacing(2);
s3.setPaintTicks(true);
s3.setPaintLabels(true);

s1.addChangeListener(this); // «уши»
s2.addChangeListener(this);
s3.addChangeListener(this);
add(s1); add(s2); add(s3); //добавили в панель
}
```

```
 public void stateChanged (ChangeEvent e){
   //обработчик события
   JSlider c = (JSlider) e.getSource();
   if ( !c.getValueIsAdjusting() ){
      if (c == s1) System.out.print ("Первый: ");
      else if (c == s2) System.out.print ("Второй: ");
      else System.out.print ("Третий: ");
      System.out.println (c.getValue());}
  }
}
//без проверки !c.getValueIsAdjusting()
//будет множественное срабатывание
//при протягивании ползунка (проверьте!)
```

```
import java.awt.*;
import javax.swing.*;
public class MyFrame{
  private static void createAndShowGUI(){
     JFrame frame = new JFrame("Элементы интерфейса");
     frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
     Container myC = frame.getContentPane();
     GUI_Slider p = new GUI_Slider();
     myC.add(p);
     frame.setSize(550,300);
     frame.setLocation(10,10);
     frame.setVisible(true);
  }
  public static void main (String[ ] args){
     javax.swing.SwingUtilities.invokeLater(new Runnable(){
        public void run(){createAndShowGUI();}});
  }
}
```

BlueJ: Terminal Window - GUI_ползунок

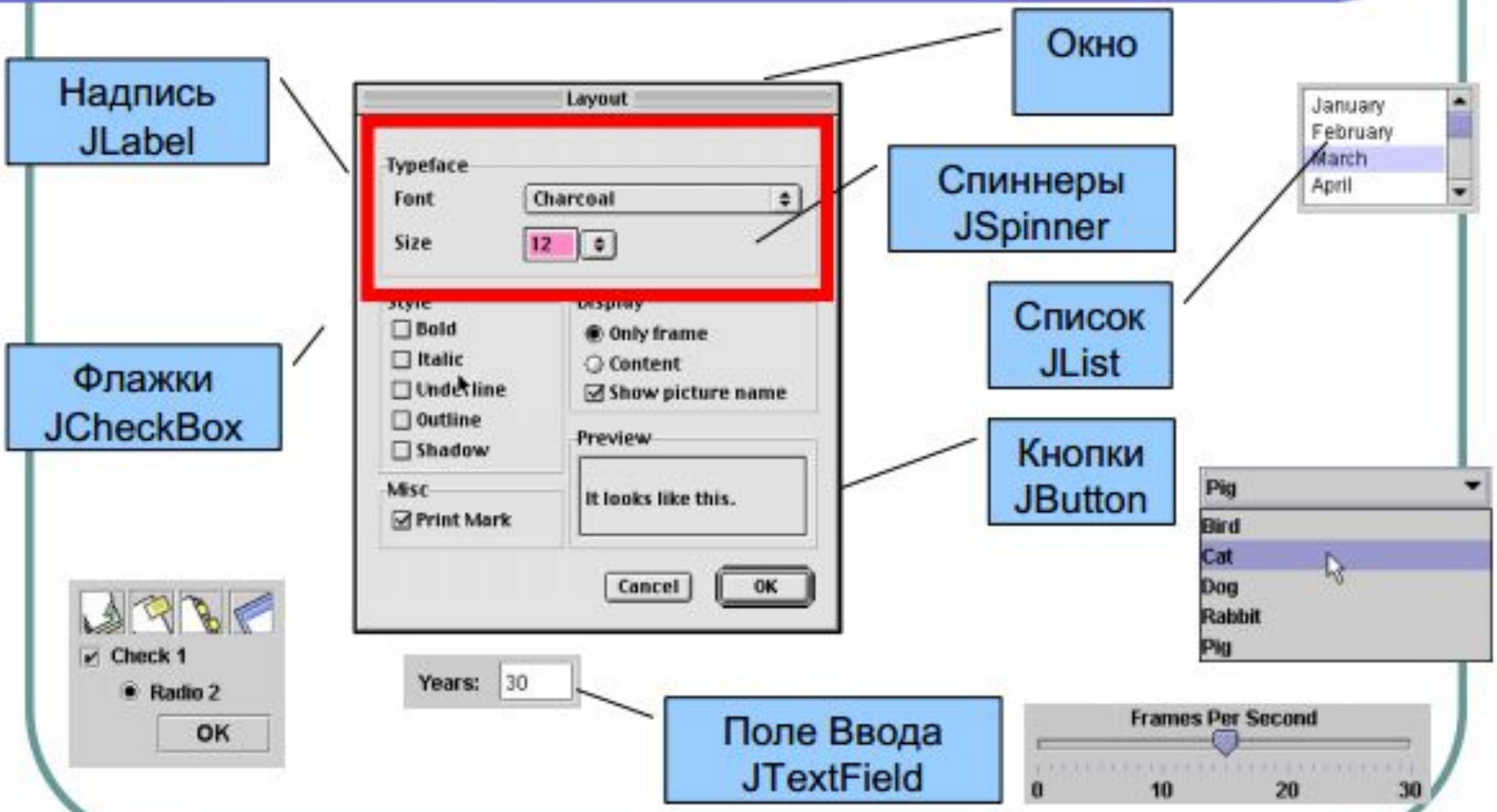
Options

```
Первый: 1
Первый: -1
Второй: 30
Третий: 73
Первый: 3
Первый: 1
Второй: 20
Третий: 29
```

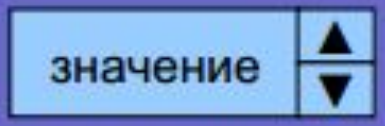
Элементы интерфейса

- Сложные элементы интерфейса (1)
  - JSpinner
  - JMenu
  - JMenuBar
- Каждый из элементов содержит другие элементы и отличается довольно сложным поведением

Элементы интерфейса (примеры)



JSpinner – "Колесико"



- Функционально – похож на JList и JComboBox
- Элемент позволяет задать значение некоторого параметра непосредственно (в поле ввода) или щелчками по кнопкам "меньше" "больше"
- Реализация элемента опирается на ряд понятий:
  - модель данных
  - способы форматирования и редактирования
  - изменение состояния и реакция на него

Вспомогательные классы JSpinner (1)

- Модели
  - SpinnerModel (по умолчанию –
  - AbstractSpinnerModel (баз. класс моделей)
  - SpinnerListModel (список или массив)
  - SpinnerNumberModel (н.у., мин, макс, шаг)
  - SpinnerDateModel (-- " --, только даты)
  - Модель пользователя

События JSpinner и реакция на них

- изменение значения в поле (вводом или кнопками) →событие ChangeEvent
- для реакции:
  - реализовать интерфейс ChangeListener – метод stateChanged (ChangeEvent e)
  - зарегистрировать слушателя события – addChangeListener(…)

Событие возникает при изменении значения, т.е. при щелчке по кнопке «больше» («меньше»), а также при нажатии на клавишу «Enter» после ввода в текстовое поле нового значения, отличающегося от предыдущего.

Вспомогательные классы JSpinner (2)

- Редакторы
  - JSpinner.DefaultEditor
  - JSpinner.NumberEditor (работа с последовательностями чисел)
  - JSpinner.ListEditor (работа с элементами списка или массива)
  - JSpinner.DateEditor (работа с последовательностями дат)
- Стандартная модель – редактор выбирается автоматически. Доступ к полю – Read / Write

Конструкторы JSpinner

- JSpinner( ) конструктор создает спиннер с моделью int SpinnerNumberModel, НУ=0, мин=-∞, макс= ∞, шаг=1
- JSpinner(SpinnerModel) параметр указывает собственную модель с необходимыми НУ, мин, макс, шаг

Методы JSpinner

- void setValue(Object)
  Object getValue( ) задать / получить значение элемента из поля
- Object getNextValue( )
  Object getPreviousValue( ) получить зн. следующего / предыдущего элемента (относительно getValue())
- SpinnerModel getModel( )
  void setModel(SpinnerModel) получить / задать модель
- JComponent getEditor( )
  void setEditor(JComponent) получить / задать редактор
- protected JComponent createEditor(SpinnerModel)
  (переопределяется, если применяется нестандартная модель)

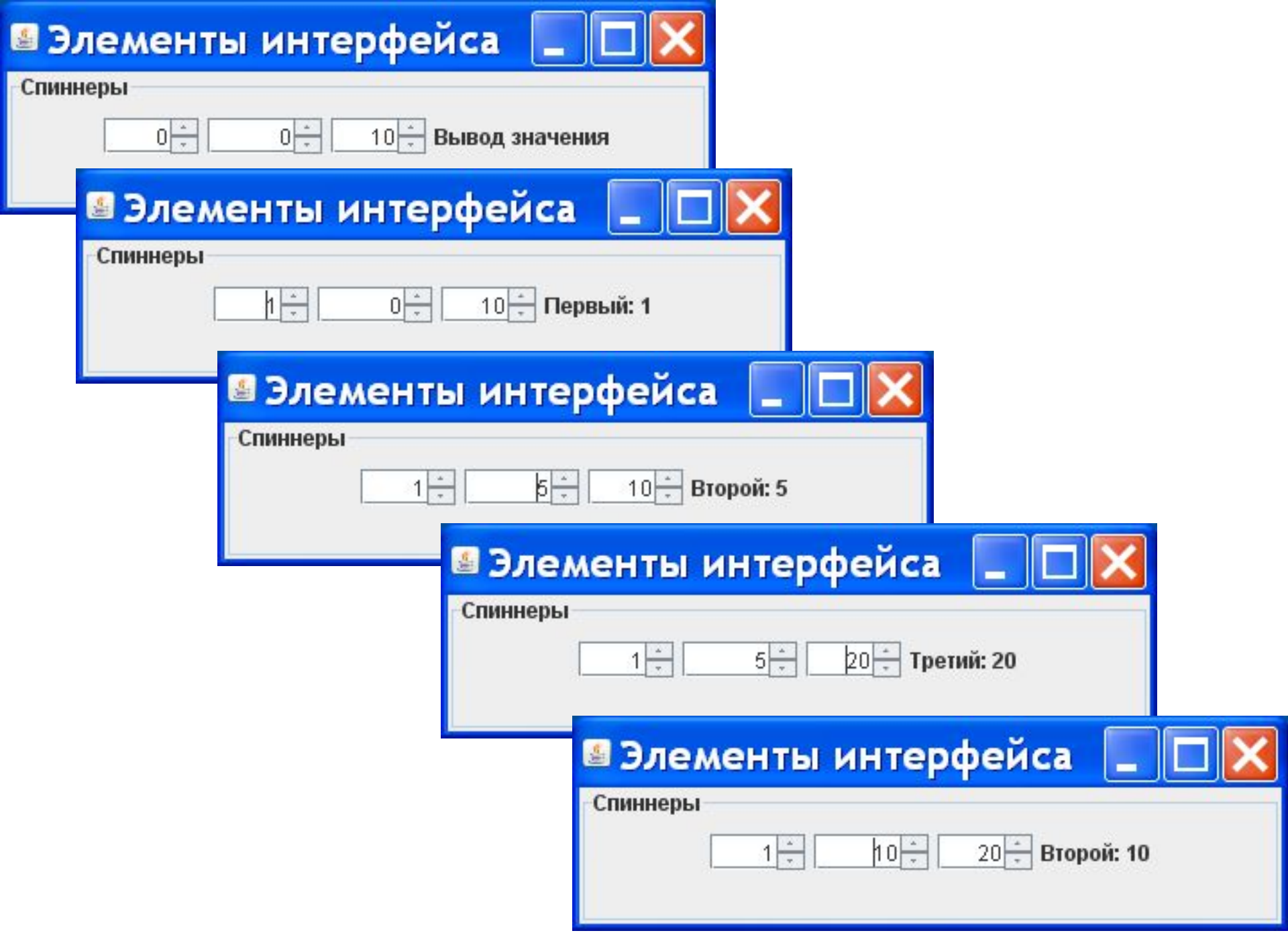
JSpinner – пример 1

- Создаются три числовых спиннера с различными НУ, мин, макс, шаг
- Регистрируется слушатель события ChangeListener
- Создается метод обработки события stateChanged

```
import java.awt.*; import javax.swing.*; //Проект - 1 GUI_JSpinner_1
import java.awt.event.*; import javax.swing.event.*;
public class GUI_Spinner extends JPanel
                                    implements ChangeListener {
   JSpinner s1,s2,s3;
   JLabel lab;
   public GUI_Spinner(){ // конструктор
     SpinnerModel  numModel1 = new SpinnerNumberModel(0,-10,10,1);
     s1 = new JSpinner (numModel1);
     SpinnerModel numModel2 = new SpinnerNumberModel(0,-100,100,5);
     s2 = new JSpinner (numModel2);
     SpinnerModel numModel3 = new SpinnerNumberModel(10,0,220,10);
     s3 = new JSpinner (numModel3);
     lab = new JLabel("Вывод значения");
     // добавляем слушателя (текущую панель) к источникам событий
     s1.addChangeListener(this);
     s2.addChangeListener(this);
     s3.addChangeListener(this);
     //добавляем спиннеры  и метку  на текущую панель
     add (s1);   add (s2);   add (s3);   add (lab);
     //панель будет иметь название
     setBorder (new  javax.swing.border.TitledBorder ("Спиннеры"));
     //менеджер компоновки для панели:
     setLayout (new FlowLayout ()); }
```

```
public void stateChanged (ChangeEvent e){ // обработчик
   String str;
   JSpinner  c =  (JSpinner) e.getSource(); //источник
   if (c == s1) str = "Первый: ";
   else if (c == s2) str = "Второй: ";
   else str = "Третий: ";
   lab.setText(str + c.getValue());
   }
}
```

```
import java.awt.*; import javax.swing.*;
public class MyFrame{ //интерфейс
  private static void createAndShowGUI(){
     JFrame frame = new JFrame("Элементы интерфейса");
     frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
     Container myC = frame.getContentPane();
     GUI_Spinner s = new GUI_Spinner(); //панель со спиннерами
     myC.add(s); //добавили панель к панели контента фрейма
     frame.setSize(400,120);
     frame.setLocation(10,10);
     frame.setVisible(true);
  }
  public static void main (String[ ] args){
     javax.swing.SwingUtilities.invokeLater(new Runnable(){
        public void run(){createAndShowGUI();}});
  }
}
```


Элементы интерфейса
Спиннеры
0
0
10
Вывод значения
Элементы интерфейса
Спиннеры
1
0
10
Первый: 1
Элементы интерфейса
Спиннеры
1
5
10
Второй: 5
Элементы интерфейса
Спиннеры
1
5
20
Третий: 20
Элементы интерфейса
Спиннеры
1
10
20
Второй: 10

Пример 2 – Спиннер с массивом

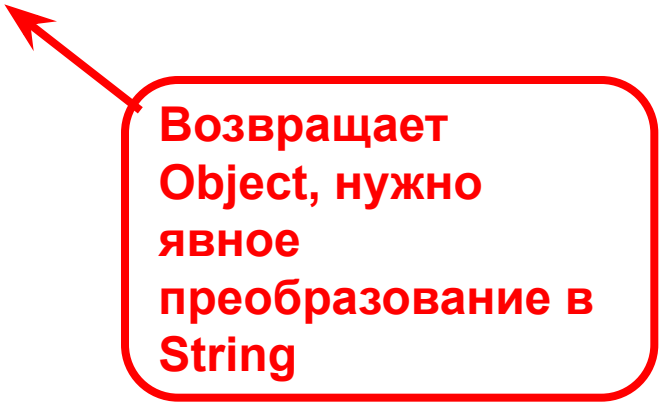
- Будет создано 2 спиннера с моделями SpinnerListModel
- Данными для одной модели будут дни недели (массив)
- Для второй модели данными будут названия месяцев (массив)

```
import java.awt.*; import javax.swing.*;
import java.awt.event.*; import javax.swing.event.*;
// Проект - 2 GUI_JSpinner_2
public class GUI_Spinner2 extends JPanel
                implements ChangeListener {
   //поля класса
   JSpinner s1,s2;
   JLabel lab1, lab2;
   String [ ] months = {"январь","февраль","март",
                        "апрель","май","июнь","июль",
                        "август","сентябрь","октябрь",
                        "ноябрь","декабрь"};
   String [ ] days = {"понедельник","вторник","среда",
                      "четверг","пятница","суббота",
                      "воскресенье"};
```

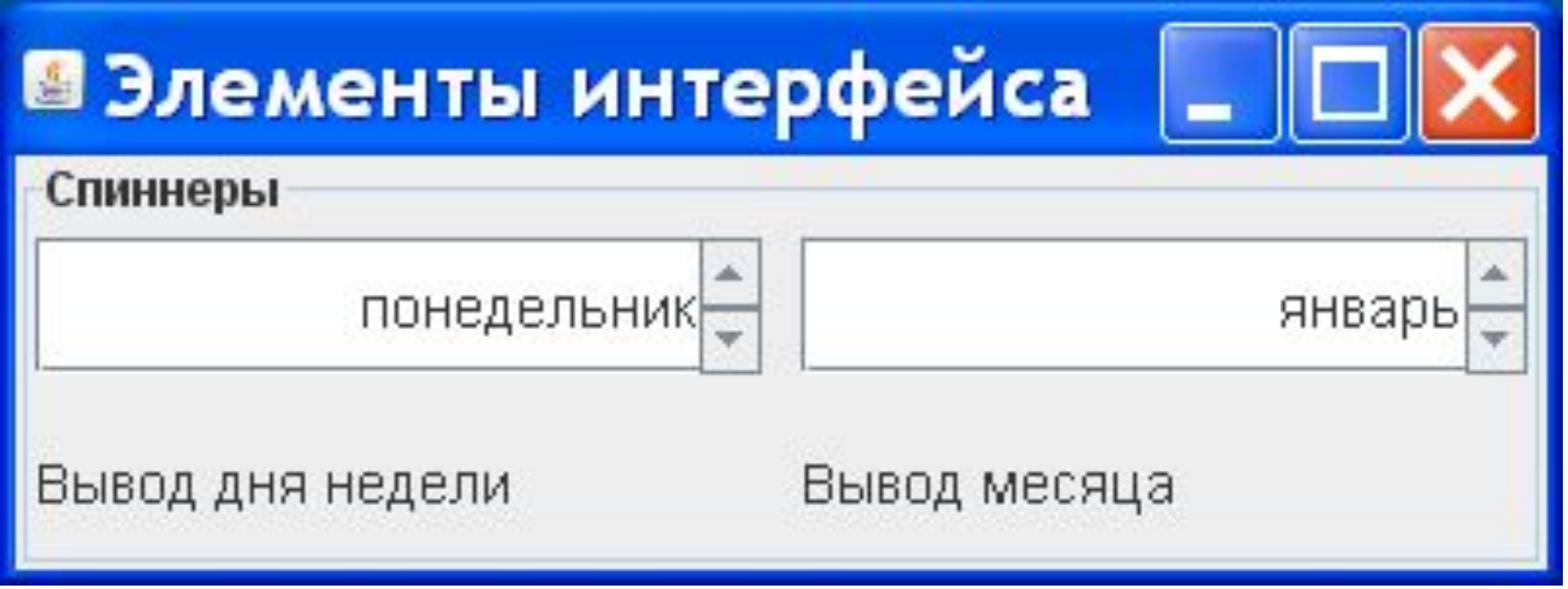
```
public GUI_Spinner2(){ //конструктор
  Font f = new Font("Ariel",Font.PLAIN,14);

  SpinnerModel  listModel1 = new  SpinnerListModel(days);
  s1 = new JSpinner (listModel1); s1.setFont(f);

  SpinnerModel  listModel2 = new SpinnerListModel(months);
  s2 = new JSpinner (listModel2); s2.setFont(f);

  lab1 = new JLabel("Вывод дня недели"); lab1.setFont(f);
  lab2 = new JLabel("Вывод месяца"); lab2.setFont(f);

 s1.addChangeListener(this); s2.addChangeListener(this);

  add (s1);   add (s2);   add (lab1); add (lab2);

  setBorder(new javax.swing.border.TitledBorder("Спиннеры"));
  setLayout(new GridLayout(2,2,10,10));  }
```

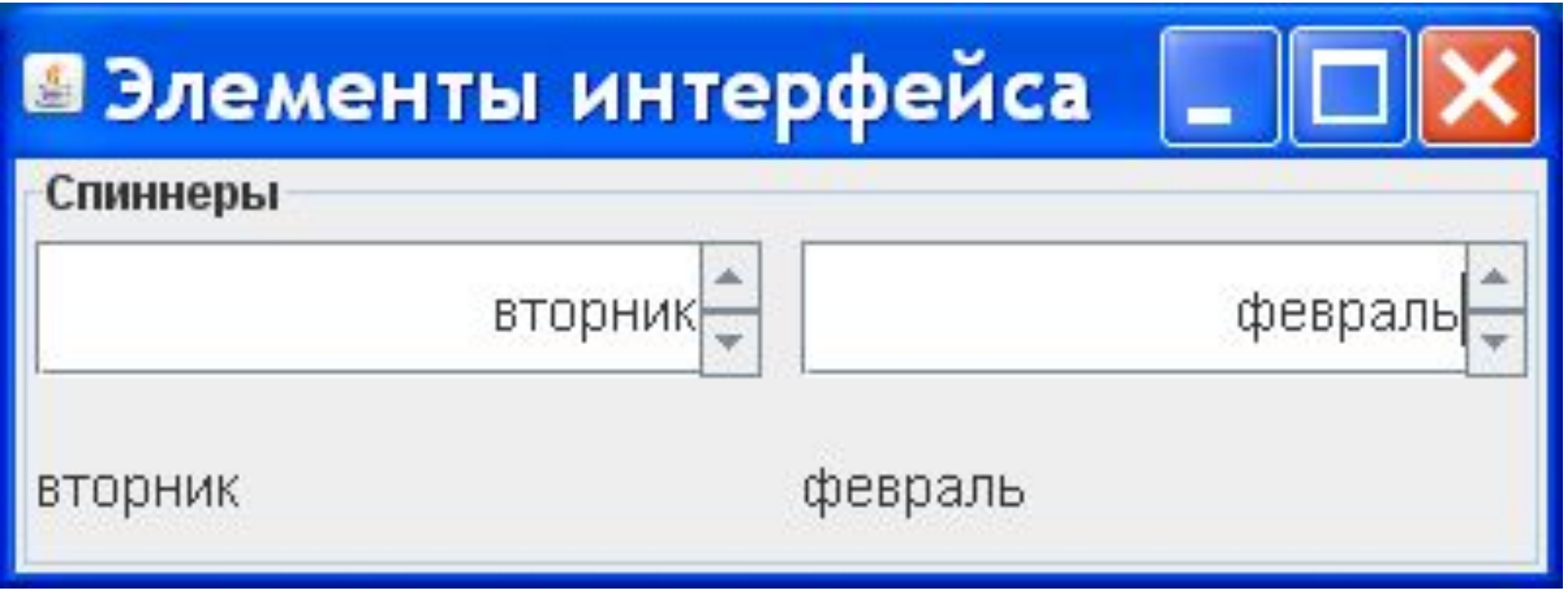
Интерфейсная ссылка

```
public void stateChanged (ChangeEvent e){ //обработчик
   String str;
   JSpinner c =  (JSpinner) e.getSource();
   if (c == s1) lab1.setText((String) c.getValue());
   else lab2.setText((String) c.getValue());
  }
}
```



```
import java.awt.*; import javax.swing.*;
public class MyFrame{ //интерфейс
  private static void createAndShowGUI(){
     JFrame frame = new JFrame("Элементы интерфейса");
     frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
     Container myC=frame.getContentPane();
     GUI_Spinner2 s= new GUI_Spinner2();
     myC.add(s);
     frame.setSize(400,150);
     frame.setLocation(10,10);
     frame.setVisible(true);
  }
  public static void main (String[ ] args){
     javax.swing.SwingUtilities.invokeLater(new Runnable(){
        public void run(){createAndShowGUI();}});
  }
}
```


Элементы интерфейса
Спиннеры
понедельник
январь
Вывод дня недели
Вывод месяца


Элементы интерфейса
Спиннеры
вторник
февраль
вторник
февраль