



# Разработка оконных приложений на Java и Swing



Top 100 features missed in Swing

# План презентации

- Почему именно Swing
- Swing Appliaction Framework –  
решение общих проблем для всех оконных приложений
- Разработка визуального интерфейса и внешний вид приложения
- Реализации логика работы приложения и его модели данных
- Решение сопутствующих проблем:  
фоновые вычисления и  
взаимодействие с окружением
- Распространение приложения



# Почему именно Swing

- Гибкий и без ограничений  
таблица на 1 000 000 записей
- Состоявшийся
- Симпатичный интерфейс
- Прост для программиста
- Легко найти документацию
- Много готовых решений
- Визуальный дизайнер
- Мультиплатформенный

# Общие проблемы при разработке Desktop приложений

## Swing Application From Scratch

```
import javax.swing.*;
public class HelloWorldSwing {
    /** Create the GUI and show it.  For thread safety,
     *  this method should be invoked from the
     *  event-dispatching thread.
     */
    private static void createAndShowGUI() {
        //Create and set up the window.
        JFrame frame = new JFrame("HelloWorldSwing");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        //Add the ubiquitous "Hello World" label.
        JLabel label = new JLabel("Hello World");
        frame.getContentPane().add(label);

        //Display the window.
        frame.pack();
        frame.setVisible(true);
    }
    public static void main(String[] args) {
        //Schedule a job for the event-dispatching thread:
        //creating and showing this application's GUI.
        javax.swing.SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                createAndShowGUI();
            }
        });
    }
}
```

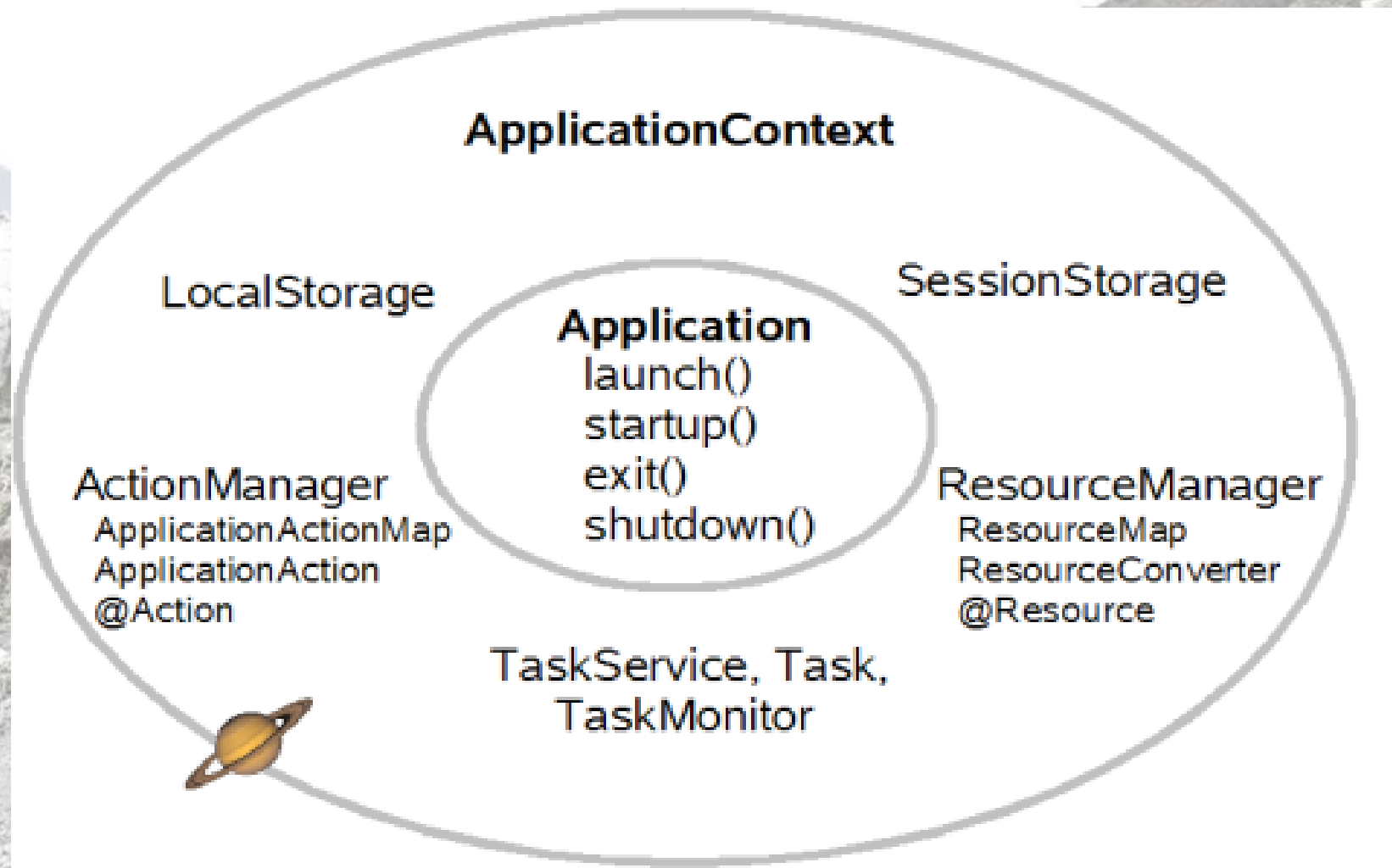
# Общие проблемы swing/Desktop приложения:

- Управление жизненным циклом приложения (инициализация, запуск, ..., завершение)
- Локализация и управление ресурсами приложения
- Обработка событий, управление событиями
- Управление задачами («вычислениями»)
- Сохранения состояния приложения (положения форм, состояния компонент, профиль пользователя)



# Swing Application Framework

- <https://appframework.dev.java.net/>
- <https://appframework.dev.java.net/intro/index.html>
- <http://java.sun.com/developer/technicalArticles/javase/swingappfr/>



# Жизненный цикл приложения

- Initialize
- Startup
- Ready
- do exit
- shutdown





# Жизненный цикл приложения: подробно

initialize	С вызовом метода передаются аргументы командной строки приложения. Здесь можно подготовить невизуальную конфигурацию приложения.
startup	Непосредственно запуск приложения, конфигурация панелей приложения и прочее
ready	Вызывается после того как метод startup отработал, в момент вызова этого метода, метод <code>Application.getInstance(MyApp.class)</code> вернёт корректный экземпляр вашего приложения. Если вызвать его раньше — можно получить неконфигурированную пустышку и поломать тем самым приложение, поскольку результат метода кешируется.
exit	Вы сами вызываете этот метод в случае когда приложения пора завершиться. При этом Framework опрашивает слушателей о возможности выхода из приложения и есть возможность отклонить выход. Смотри пример со слушателем.
shutdown	Метод вызывается перед выходом из приложения. Реализация по умолчанию сохраняет состояния всех компонент, так что при повторном запуске приложения оно будет выглядеть так же как и до выключения. Переопределив метод можно добавить собственную функциональность.
<see docs>	Есть и другие методы вызываемые при запуске и конфигурации приложения, подробнее смотри документация. (например <code>configureWindow</code> )





# Пример приложения на SwingAppFramework

```
public class BasicFrameworkApp extends Application {
    private JFrame mainFrame;
    private JLabel label;

    @Override
    protected void startup() {
        mainFrame = new JFrame("BasicFrameworkApp");
        mainFrame.setDefaultCloseOperation(JFrame.DO_NOTHING_ON_CLOSE);
        mainFrame.addWindowListener(new WindowAdapter() {
            public void windowClosing(WindowEvent e) {
                mainFrame.setVisible(false);
                exit();
            }
        });
        label = new JLabel("Hello, world!");
        mainFrame.add(label);
        mainFrame.pack();
        mainFrame.setVisible(true);
    }

    public static void main(String[] args) {
        Application.launch(BasicFrameworkApp.class, args);
    }
}
```

# Пример SingleFrame приложения

```
public class BasicSingleFrameApp extends SingleFrameApplication {
    JLabel label;

    @Override
    protected void startup() {
        getMainFrame().setTitle("BasicSingleFrameApp");
        label = new JLabel("Hello, world!");
        label.setFont(new Font("SansSerif", Font.PLAIN, 22));
        show(label);
    }

    public static void main(String[] args) {
        Application.launch(BasicSingleFrameApp.class, args);
    }
}
```





# Пример, когда слушатель выхода из приложения может запретить выход

```
@Override
protected void startup() {
    getMainFrame().setTitle("ConfirmExit");
    exitButton = new JButton("Exit Application");
    exitButton.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
            exit(e);
        }
    });
    addExitListener(new ExitListener() {
        public boolean canExit(EventObject e) {
            boolean bOkToExit = false;
            Component source = (Component) e.getSource();
            bOkToExit = JOptionPane.showConfirmDialog(source,
                "Do you really want to exit?" ==
                JOptionPane.YES_OPTION;
            return bOkToExit;
        }
        public void willExit(EventObject event) {

        }
    });
    show(exitButton);
}
```

# Контекст приложения:

Контекст приложения предоставляет доступ ко всей функциональности фреймворка через набор статических методов.

TODO





# Локализация и управление ресурсами приложения

```
public class HelloWorld extends SingleFrameApplication {
    JLabel label;
    ResourceMap resource;

    @Override
    protected void initialize(String[] args) {
        ApplicationContext ctxt = getContext();
        ResourceManager mgr = ctxt.getResourceManager();
        resource = mgr.getResourceMap(HelloWorld.class);
    }

    @Override
    protected void startup() {
        label = new JLabel();
        String helloText = (String) resource.getObject("helloLabel", String.class);
        // Or you can use the convenience methods that cast resources
        // to the type indicated by the method names:
        // resource.getString("helloLabel.text");
        // resource.getColor("backgroundcolor");
        // and so on.
        Color backgroundColor = resource.getColor("color");
        String title = resource.getString("title");
        label.setBackground(backgroundColor);
        label.setOpaque(true);
        getMainFrame().setTitle(title);
        label.setText(helloText);
        show(label);
    }
    // ...
}
```

# Виды ресурсов приложения

Тип результата	Имя метода ResourceMap
java.awt.Font	<a href="#">getFont</a> (java.lang.String key)
javax.swing.Icon	<a href="#">getIcon</a> (java.lang.String key)
java.lang.String	<a href="#">getString</a> (java.lang.String key, java.lang.Object... args)

# Место хранения файлов с ресурсами и правила именования

Имя класса	Имя ресурса	Имя файла
demo.MyApp	demo.resources.MyApp	demo/resources/MyApp.properties demo/resources/MyApp_en.properties demo/resources/MyApp_ru.properties





# Actions - Действия

```
// ctx is the ApplicationContext instance.  
ActionMap map = ctxt.getActionMap(this);  
  
btnMakeLarger.setAction(map.get("makeLarger"));  
btnMakeSmaller.setAction(map.get("makeSmaller"));  
  
# resources/ResizeFontPanel.properties  
makeLarger.Action.text = Increase font size  
makeLarger.Action.icon = increase.png  
  
makeSmaller.Action.text = Decrease font size  
makeSmaller.Action.icon = decrease.png
```



# Состояние приложение и профиль пользователя

- LocalStorage
- XMLEncode | XMLDecoder
- Каждый пользователь имеет собственный профиль

## Пример доступа к локальному хранилищу

```
@Action
public void loadMap() throws IOException {
    Object map = ctxt.getLocalStorage().load(file);
    listModel.setMap((LinkedHashMap<String, String>)map);
    showFileMessage("loadedFile", file);
}

@Action
public void saveMap() throws IOException {
    LinkedHashMap<String, String> map = listModel.getMap();
    ctxt.getLocalStorage().save(map, file);
    showFileMessage("savedFile", file);
}
```

## Место хранения файлов различно для ОС, например в Windows это:

```
${userHome}\Application Data\${vendorId}\${applicationId}\session.xml
```



# SingleFrameApplication сохраняет состояние интерфейса по умолчанию

По умолчанию SingleFrameApplication обеспечивает сохранения состояния основных элементов интерфейса приложения. Позиции и размеры окон, выбранные вкладки в таббаре, положения разделителя в JsplitPane и прочее. Это обеспечивается примерно так:

```
String sessionFile = "sessionState.xml";
ApplicationContext ctx = getContext();
JFrame mainFrame = getMainFrame();

@Override protected void startup() {
    //...
    /* Restore the session state for the main frame's component tree.
     */
    try {
        ctx.getSessionStorage().restore(mainFrame, sessionFile);
    }
    catch (IOException e) {
        logger.log(Level.WARNING, "couldn't restore session", e);
    }
    // ...
}

@Override protected void shutdown() {
    /* Save the session state for the main frame's component tree.
     */
    try {
        ctx.getSessionStorage().save(mainFrame, sessionFile);
    }
    catch (IOException e) {
        logger.log(Level.WARNING, "couldn't save session", e);
    }
}
```

# Визуальные компоненты

## «Пропущенные» компоненты: Календарь

- <http://www.toedter.com/en/jcalendar/index.html>

<div>May 2006</div> <table><tr><td>Sun</td><td>Mon</td><td>Tue</td><td>Wed</td><td>Thu</td><td>Fri</td><td>Sat</td></tr><tr><td>18</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td></tr><tr><td>19</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td></tr><tr><td>20</td><td>14</td><td>15</td><td>16</td><td>17</td><td>18</td><td>19</td></tr><tr><td>21</td><td>21</td><td>22</td><td>23</td><td>24</td><td>25</td><td>26</td></tr><tr><td>22</td><td>28</td><td>29</td><td>30</td><td>31</td><td></td><td></td></tr></table>	Sun	Mon	Tue	Wed	Thu	Fri	Sat	18	1	2	3	4	5	6	19	7	8	9	10	11	12	20	14	15	16	17	18	19	21	21	22	23	24	25	26	22	28	29	30	31			<div>17.04.2006</div> <div>17.04x.2006</div> <div>__/__/</div> <div>17.04.2006</div>	<div>2006</div>	<div>English (United States)</div>
Sun	Mon	Tue	Wed	Thu	Fri	Sat																																							
18	1	2	3	4	5	6																																							
19	7	8	9	10	11	12																																							
20	14	15	16	17	18	19																																							
21	21	22	23	24	25	26																																							
22	28	29	30	31																																									
<div>Sun Mon Tue Wed Thu Fri Sat</div> <table><tr><td>18</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td></tr><tr><td>19</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td></tr><tr><td>20</td><td>14</td><td>15</td><td>16</td><td>17</td><td>18</td><td>19</td></tr><tr><td>21</td><td>21</td><td>22</td><td>23</td><td>24</td><td>25</td><td>26</td></tr><tr><td>22</td><td>28</td><td>29</td><td>30</td><td>31</td><td></td><td></td></tr></table>	18	1	2	3	4	5	6	19	7	8	9	10	11	12	20	14	15	16	17	18	19	21	21	22	23	24	25	26	22	28	29	30	31			<div>2004x</div> <div>2004</div> <div>2004</div>	<div>May</div>								
18	1	2	3	4	5	6																																							
19	7	8	9	10	11	12																																							
20	14	15	16	17	18	19																																							
21	21	22	23	24	25	26																																							
22	28	29	30	31																																									



# «Пропущенные» компоненты: LinkLabel

- <http://www.jetbrains.com/idea/openapi/5.0/com/intellij/ui/components/labels/LinkLabel.html>
- Компонент легко реализовать самостоятельно
- Можно использовать реализацию от JetBrains

1. Создать новое исследование  
2. Задать параметры исследования  
3. Задать результаты опроса  
4. Анализ: социоматрица  
5. Анализ: социограмма  
6. Заключение

Назад | Далее ➔



# Пример: реализация LinkLabel

```
import java.awt.Color;
import java.awt.Cursor;
import java.awt.Font;
import java.awt.Graphics;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.beans.PropertyChangeEvent;
import java.beans.PropertyChangeListener;
import java.util.HashSet;
import java.util.Set;
import javax.swing.Action;
import javax.swing.Icon;
import javax.swing.JLabel;

/**
 * Label which behaves like hyperlink.
 *
 * @author dmitry.mamonov
 */
public class LinkLabel extends JLabel {
    Set<ActionListener> listeners = new HashSet<ActionListener>();
    boolean highlighted = false;
    boolean selected = false;
    boolean highlightBackground = false;
    Action action = null;
    PropertyChangeListener actionPropertyChangeListener = new PropertyChangeListener() {
        public void propertyChange(PropertyChangeEvent evt) {
            if (evt != null && "enabled".equals(evt.getPropertyName()) == true) {
                Boolean newValue = (Boolean) evt.getNewValue();
                if (newValue != isEnabled()) {
                    setEnabled(newValue);
                }
            }
        }
    };

    public LinkLabel() {
        super();
        setForeground(new Color(0x0000CC));
        setBackground(new Color(0xC3D9FF));
    }
}
```

```

setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
addMouseListener(new MouseAdapter() {
    @Override
    public void mousePressed(MouseEvent e) {
        ActionEvent event = new ActionEvent(this, 0, "clicked");
        for (ActionListener lst : listeners) {
            lst.actionPerformed(event);
        }
        if (action != null) {
            action.actionPerformed(event);
        }
    }
    @Override
    public void mouseEntered(MouseEvent e) {
        highlited = true;
        repaint();
    }
    @Override
    public void mouseExited(MouseEvent e) {
        highlited = false;
        repaint();
    }
});
}

public boolean isSelected() {
    return selected;
}

public void setSelected(boolean selected) {
    boolean oldSelected = this.selected;
    this.selected = selected;
    if (oldSelected != selected) {
        repaint();
    }
}

public boolean isHighlightBackground() {
    return highlightBackground;
}

public void setHighlightBackground(boolean highlightBackground) {
    this.highlightBackground = highlightBackground;
}

public Action getAction() {
    return action;
}

```

```

public void setAction(Action _action) {
    if (this.action != null) {
        this.action.removePropertyChangeListener(actionPropertyChangeListener);
    }
    this.action = _action;
    action.addPropertyChangeListener(actionPropertyChangeListener);
    setText((String) action.getValue(Action.NAME));
    setIcon((Icon) action.getValue(Action.SMALL_ICON));
    setToolTipText((String) action.getValue(Action.SHORT_DESCRIPTION));
}

public void addActionListener(ActionListener lst) {
    listeners.add(lst);
}

public void removeActionListener(ActionListener lst) {
    listeners.remove(lst);
}

@Override
public void paint(Graphics g) {
    int width = getWidth();
    int height = getHeight();
    if (highlightBackground == true) {
        if ((highlited && isEnabled()) || selected) {
            Color selectedBackground = getBackground();
            if (highlited && !selected) {
                int addToColor = 15;
                selectedBackground = new Color(
                    Math.min(255, selectedBackground.getRed() + addToColor),
                    Math.min(255, selectedBackground.getGreen() + addToColor),
                    Math.min(255, selectedBackground.getBlue() + addToColor));
            }
            g.setColor(selectedBackground);
            g.fillRect(0, 0, width, height);
        }
    }
    super.paint(g);
    if (highlited && isEnabled()) {
        int baseline = getBaseline(width, height) + 1;
        g.setColor(getForeground());
        g.drawLine(0, baseline, width, baseline);
    }
}
}

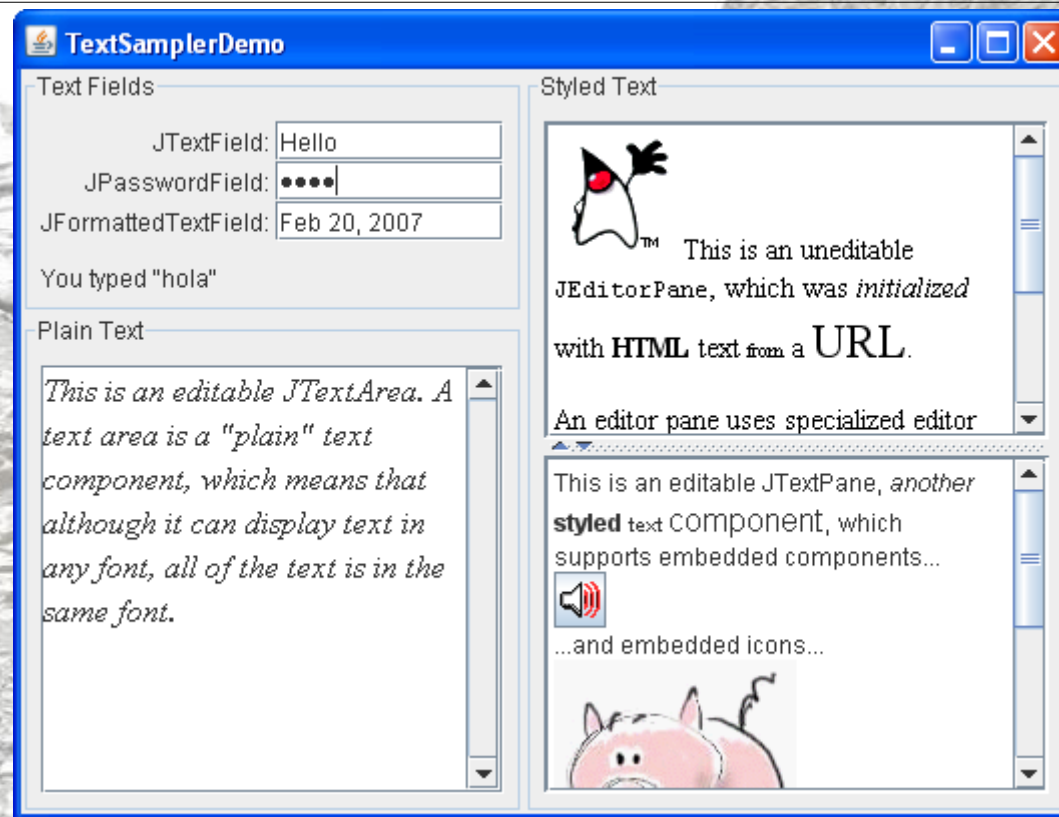
```



# Отображение HTML

- <http://java.sun.com/docs/books/tutorial/uiswing/components/editorpane.html>
- <http://java.sun.com/j2se/1.4.2/docs/api/javafx/swing/event/HyperlinkListener.html>

- JTextPane
- Гиперссылки



# Диалог выбора директории

- <http://java.sun.com/docs/books/tutorial/uiswing/components/filechooser.html>

```
import javax.swing.*;
import java.io.*;

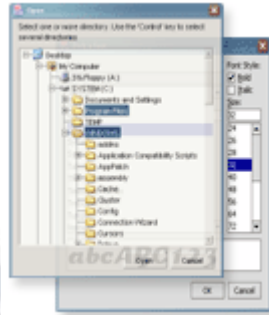
public class FC {
    public static void main(String[] args) {
        JFileChooser chooser = new JFileChooser();
        chooser.setFileSelectionMode(JFileChooser.DIRECTORIES_ONLY);
        if (JFileChooser.APPROVE_OPTION == chooser.showDialog(null,
"Select")) {
            File dir = chooser.getSelectedFile();
            System.out.println(dir);
        }
        System.exit(0);
    }
}
```



# L2FProd Components

- <http://www.l2fprod.com/common/>

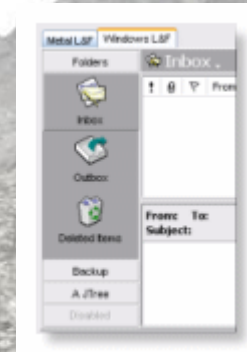
Требуются не так часто, но неплохо иметь их ввиду.



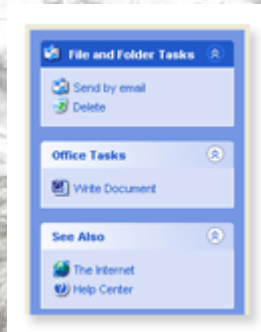
Диалоги для выбора шрифта и директории



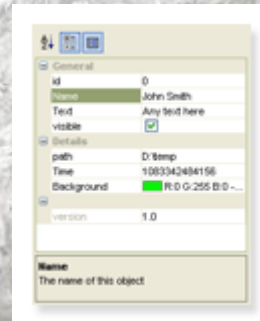
Вертикальная панель с кнопками



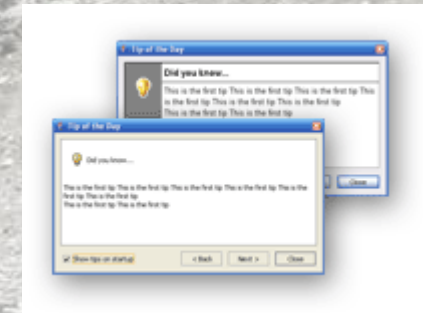
Аккордеон



Панель зачать с раскрывающимися элементами



Панель редактирования свойств бинов

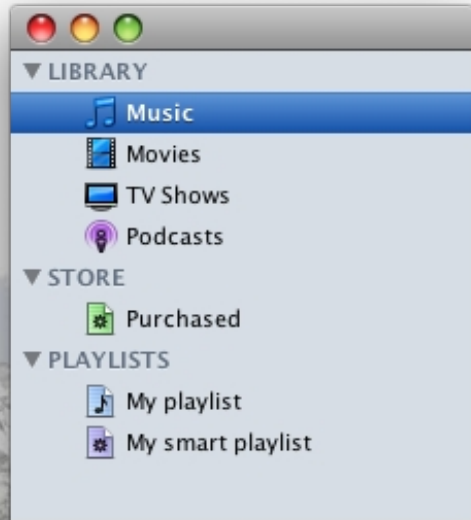


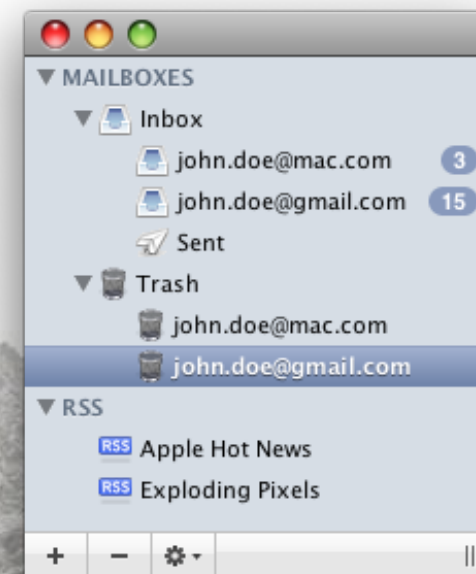
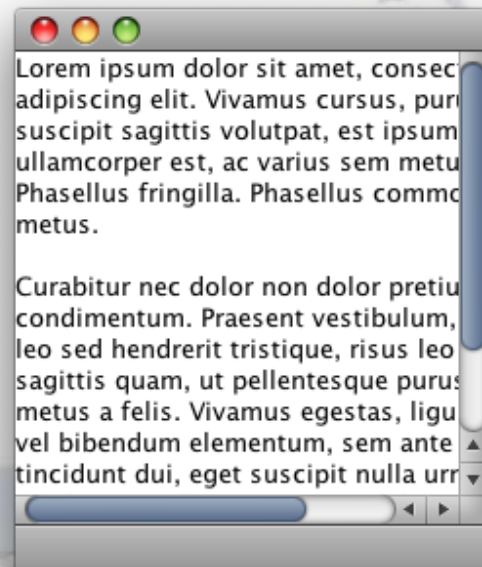
Диалог с подсказками



# Виджеты в стиле Mac OS

- <http://code.google.com/p/macwidgets/wiki/Examples>









Name	Artist	Album	
All These Things I Hate (...)	Bullet For My V...	The Poison	
Cries In Vain	Bullet For My V...	The Poison	
The End	Bullet For My V...	The Poison	
Her Voice Resides	Bullet For My V...	The Poison	
Hit The Floor	Bullet For My V...	The Poison	
Intro	Bullet For My V...	The Poison	
The Poison	Bullet For My V...	The Poison	
Room 409	Bullet For My V...	The Poison	
Spit You Out	Bullet For My V...	The Poison	
Suffocating Under Word...	Bullet For My V...	The Poison	
Tears Don't Fall	Bullet For My V...	The Poison	
4 Words (To Choke Upon)	Bullet For My V...	The Poison	
10 Years Today	Bullet For My V...	The Poison	

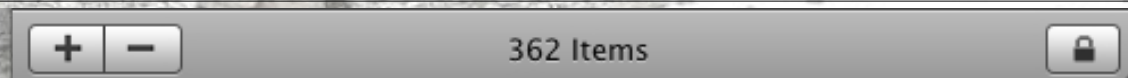
Button

☒ Check Box

Item One ▾

Label

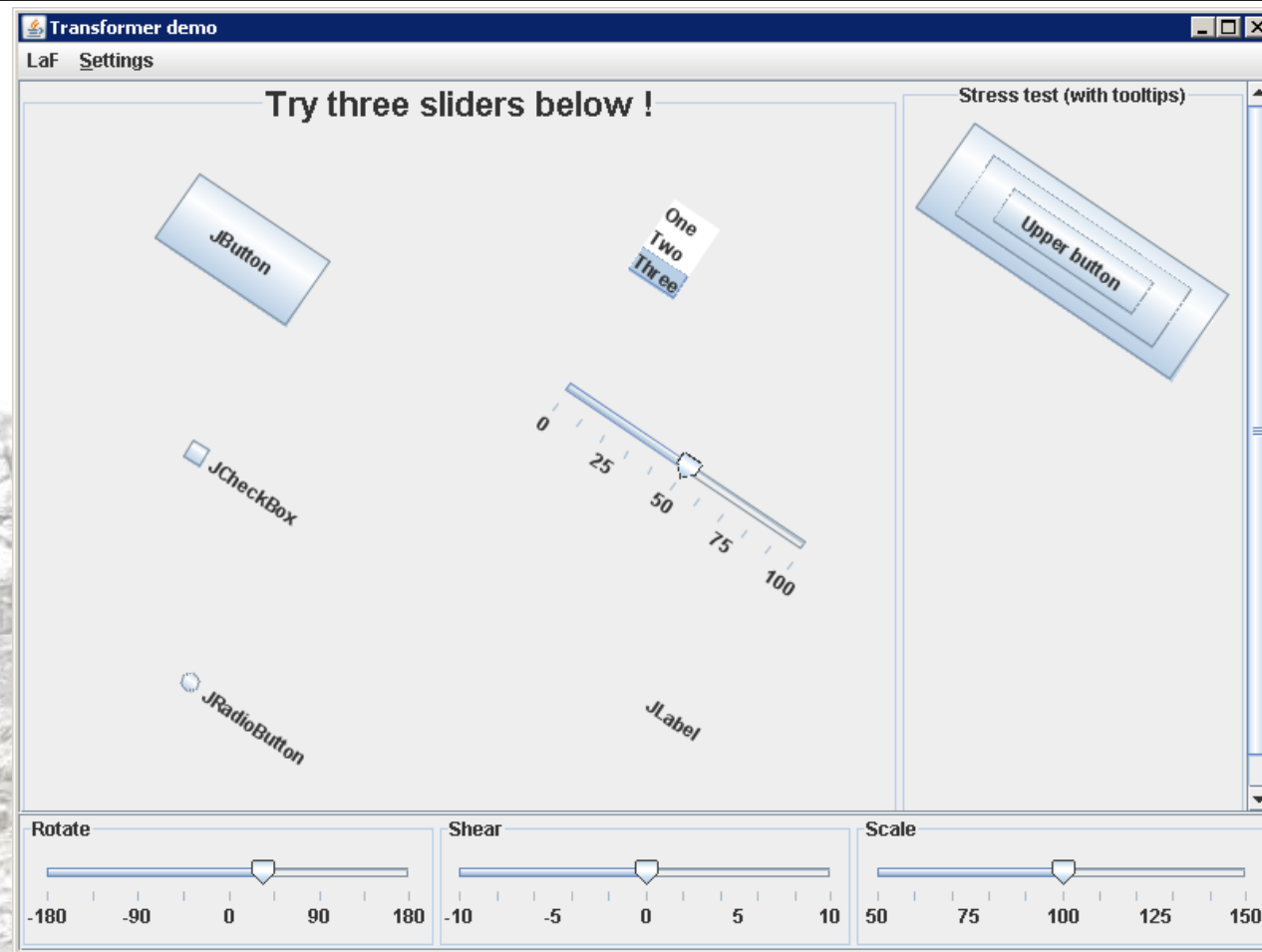
Text field





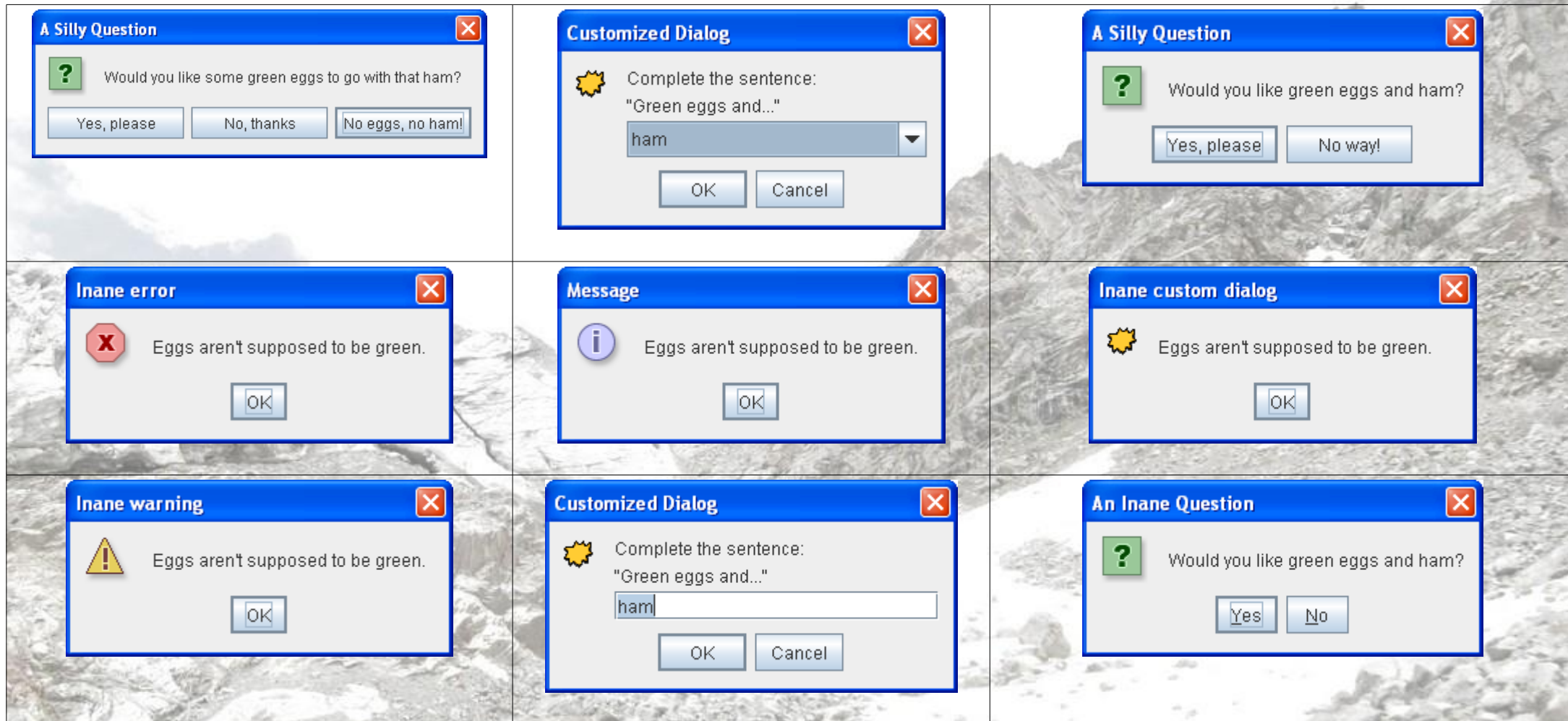
# SwingHelper: Скорее интересная чем полезная библиотека

- <https://swinghelper.dev.java.net/>



# Диалоги с сообщениями

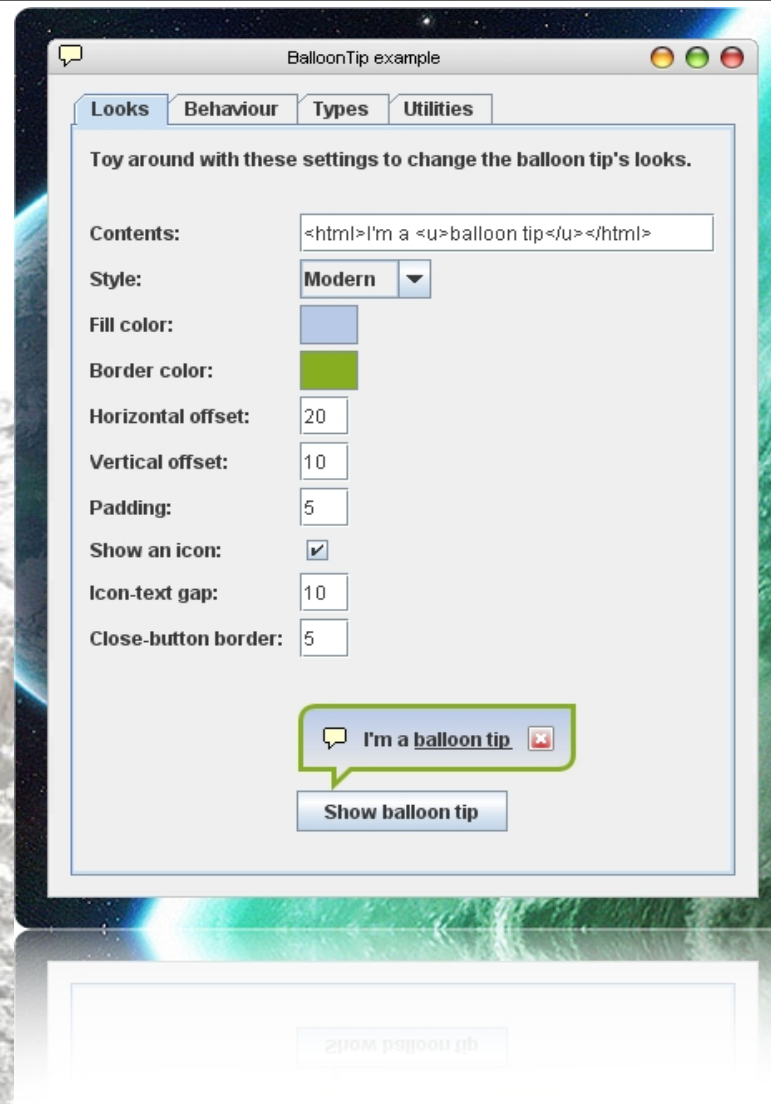
- <http://java.sun.com/docs/books/tutorial/uiswing/components/dialog.html>





# Всплывающие подсказки

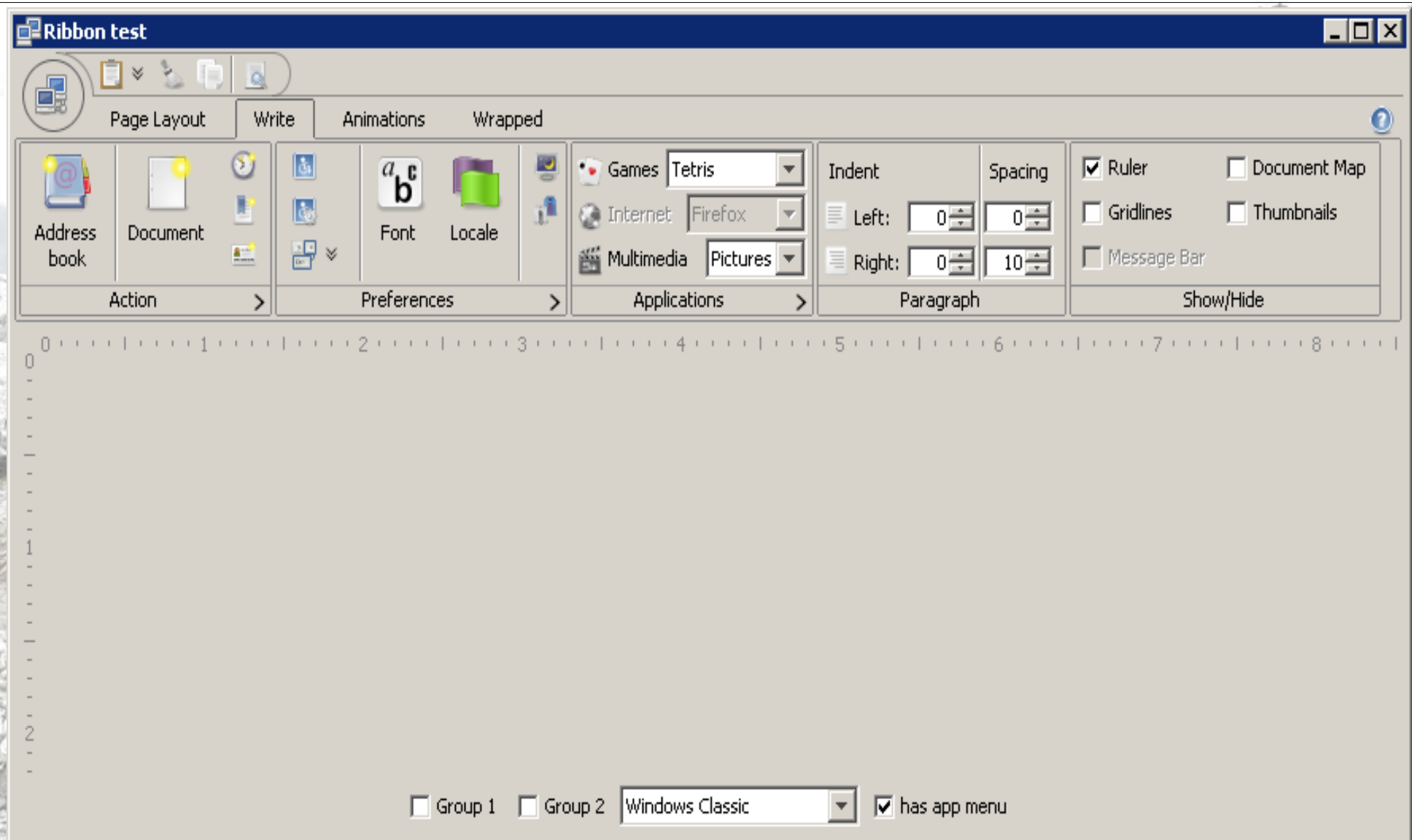
- <https://balloontip.dev.java.net/>
- <http://java.sun.com/docs/books/tutorial/uiswing/components/tooltip.html>





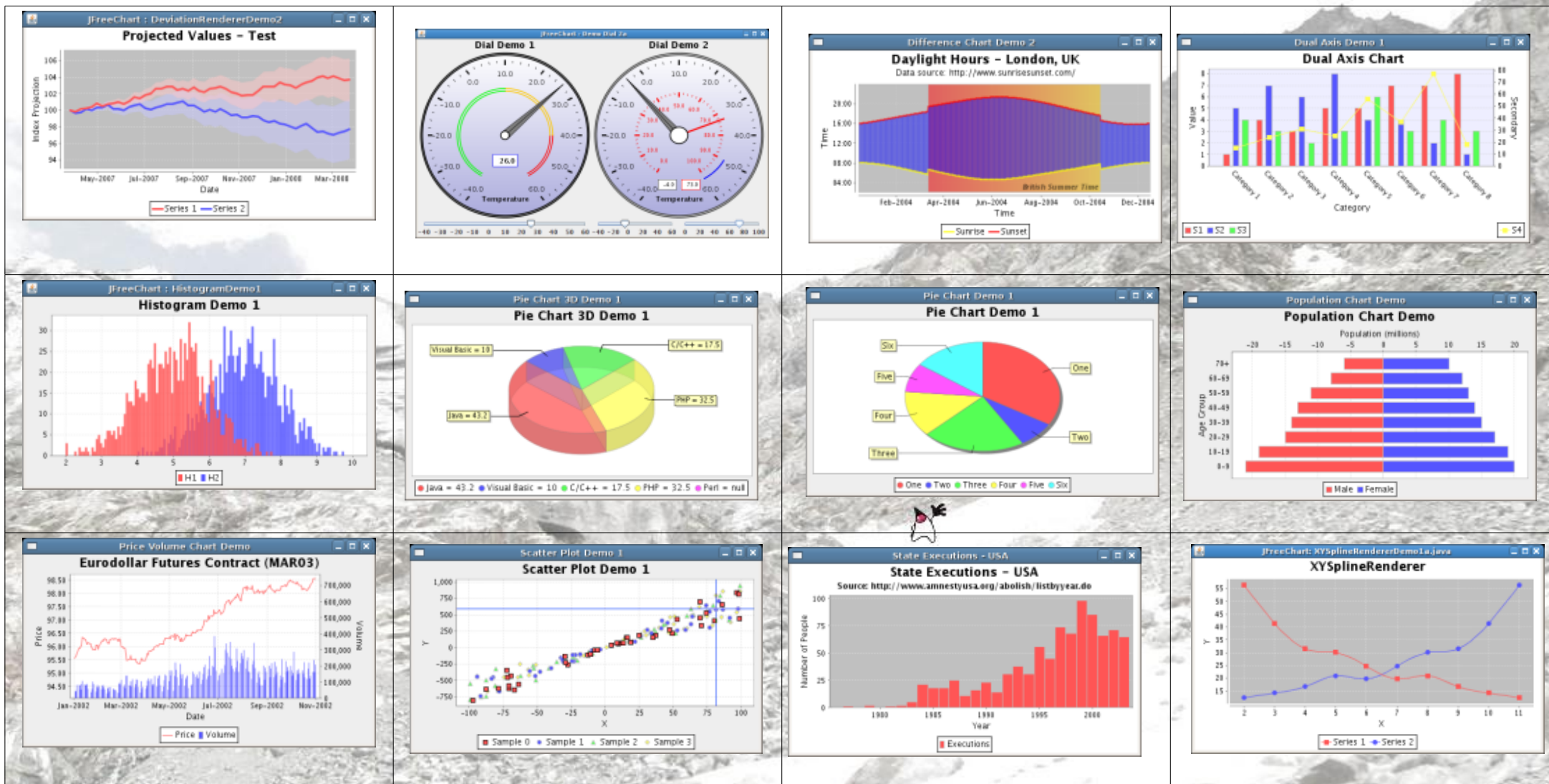
# Ribbon компонент

<https://flamingo.dev.java.net/see.html>



# Графики и диаграммы

- <http://www.jfree.org/jfreechart/samples.html>



# Использование Matisse

- <http://netbeans.org/>
- <http://netbeans.org/features/java/swing.html>

Matisse – визуальный редактор форм включённый в NetBeans IDE.

Достоинства:

- Визуальный редактор форм (поддерживает SpringLayout и другие основные менеджеры расположения компонент), выравнивание компонент по базовой линии.
- Поддерживает beansbinding для связывания данных компонент с моделью данных приложения.
- Визуальный редактор свойств компонента.
- Результат работы дизайнера — сгенерированный код, нет необходимости подключать дополнительные библиотеки или выполнять какую либо загрузку/инициализацию для его использования.
- Гибко настраиваемая генерация кода, свою логику можно вклинить практически на любом этапе.
- Может работать с нестандартными компонентами, особенно если они оформлены в виде JavaBeans.

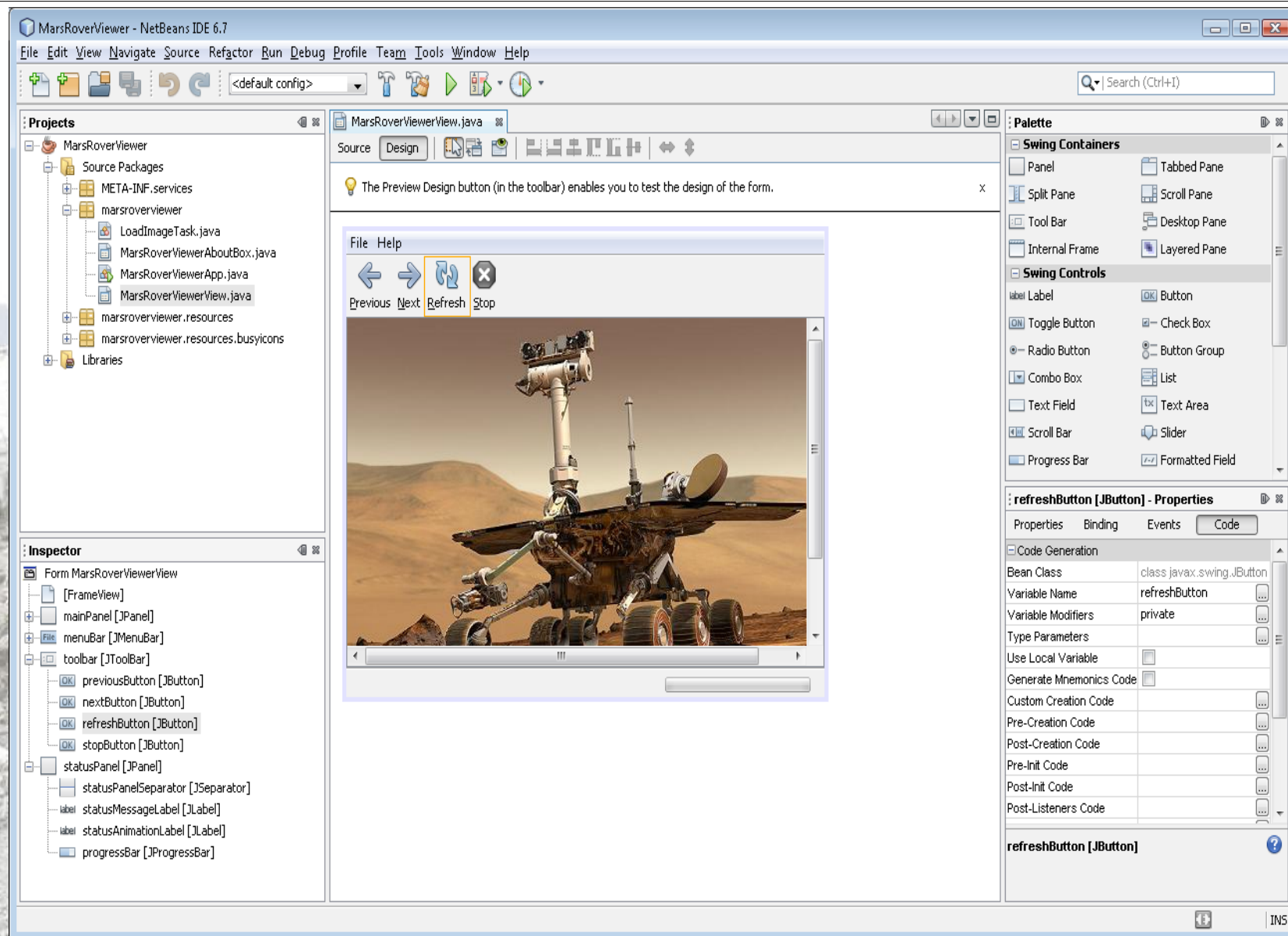
Недостатки:

- Сгенерированный код очень хорошего качества, но с самописным кодом работать проще. Хотя — кому как.
- Имеет тенденцию терять свойства компонент при рефакторинге, например горячие-клавиши для элементов меню.
- Beansbinding работает медленно.
- Форму легко поломать случайным движением мыши.

Итого, отлично подходит для разработки не слишком сложных форм и для прототипирования.



# Дизайнер форм Matisse



# Расположение компонент — LayoutManager

- <http://java.sun.com/docs/books/tutorial/uiswing/layout/using.html>
- <http://java.sun.com/docs/books/tutorial/uiswing/layout/layoutlist.html>
- <http://www.jgoodies.com/freeware/forms/index.html>
- <http://www.miglayout.com/>
- <https://tablelayout.dev.java.net/>
- <http://www.datadosen.se/riverlayout/>
- <http://pagelayout.sourceforge.net/>
- [http://en.wikipedia.org/wiki/Strategy\\_pattern](http://en.wikipedia.org/wiki/Strategy_pattern)

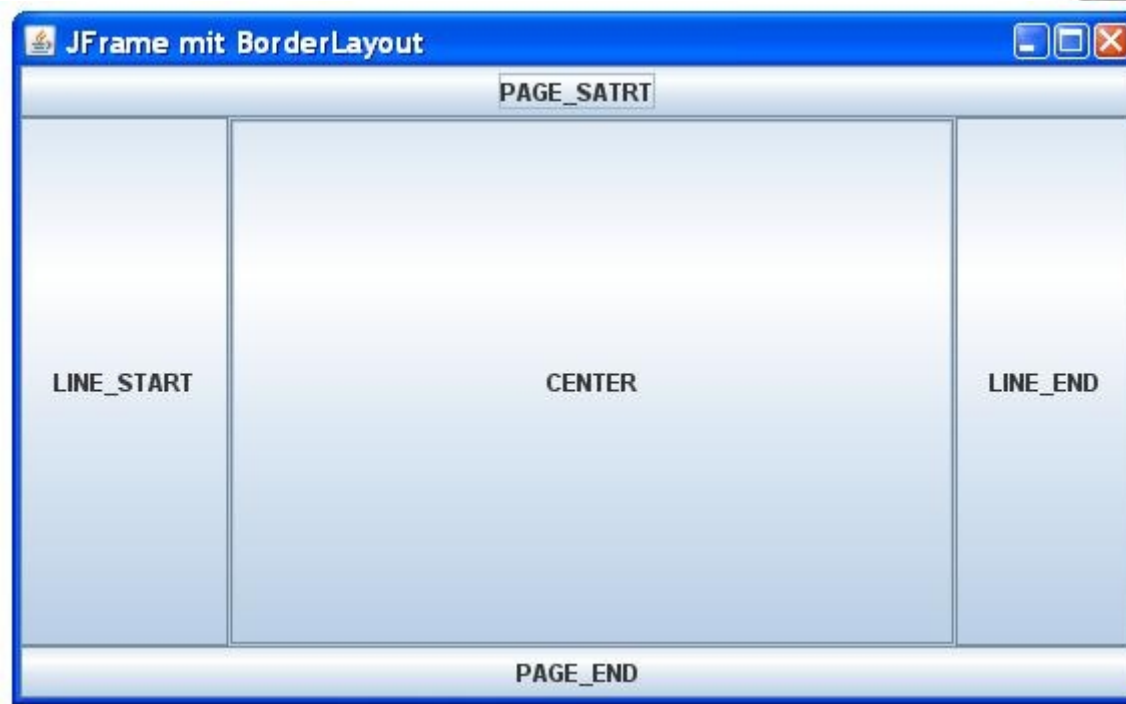
## Пример здания layout менеджера — BorderLayout:

```
JPanel panel = new JPanel();  
BoxLayout layout = new BorderLayout(panel, axis);  
panel.setLayout(layout);  
return panel;
```



# Создание форм вручную: BorderLayout

- <http://java.sun.com/docs/books/tutorial/uiswing/layout/border.html>
- <http://java.sun.com/j2se/1.4.2/docs/api/java/awt/BorderLayout.html>
- <http://javaswing.wordpress.com/2009/11/08/borderlayout/>



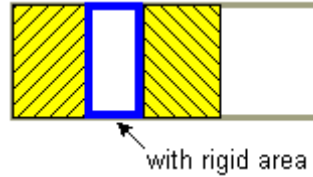


# Создание форм вручную: BorderLayout

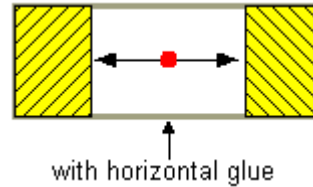
- <http://java.sun.com/docs/books/tutorial/uiswing/layout/box.html>
  - <http://java.sun.com/javase/6/docs/api/javax/swing/BoxLayout.html>
  - <http://java.sun.com/javase/6/docs/api/javax/swing/Box.html>
- Позволяет располагать компоненты в строку или столбец
  - Позволяет управлять разделителями между компонентами

# Пример работы с VoxLayout:

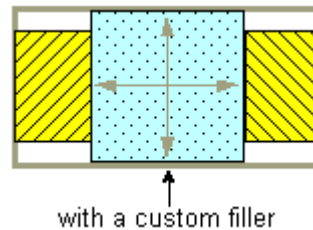
```
//Rigid area  
container.add(firstComponent);  
container.add(Box.createRigidArea(new Dimension(5,0)));  
container.add(secondComponent);
```



```
//Glue  
container.add(firstComponent);  
container.add(Box.createHorizontalGlue());  
container.add(secondComponent);
```

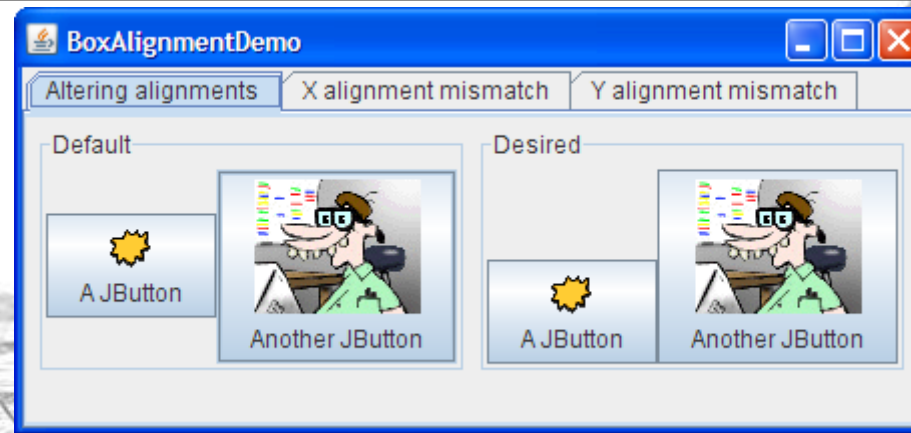


```
//Custom Box.Filler  
container.add(firstComponent);  
Dimension minSize = new Dimension(5, 100);  
Dimension prefSize = new Dimension(5, 100);  
Dimension maxSize = new Dimension(Short.MAX_VALUE, 100);  
container.add(new Box.Filler(minSize, prefSize, maxSize));  
container.add(secondComponent);
```



# Пример расположения компонент в BoxLayout

- `Jcomponent.setAlignmentX(0..1.0)`
- `Jcomponent.setAlignmentX(0..1.0)`





# Создание форм вручную FormLayout (JGoodies)

- <http://www.jgoodies.com/>

The screenshot shows a software application window titled "Sample Project - Skeleton Pro". The window has a menu bar with "File", "Components", and "Help". Below the menu bar is a toolbar with icons for file operations and a help icon. The main area is divided into four panes:

- Navigator:** A tree view showing the project structure. It includes "General Project Data", "Propeller Shaft1" (selected), "Intermediate Shaft", and "Propeller Shaft2". Each shaft has sub-items like "Segment" and "Flange".
- Dynamic Help Topics:** A list of topics including "Help", "Shaft", "Comments", and "Samples". "Comments" is currently selected.
- Propeller Shaft1 (Form):** The main form for editing the selected shaft. It contains fields for "Identifier" (Propeller Shaft1), "Power", "Speed", "Material" (C45E, ReH=600), "Ice Class" (E), and "Length". There are also text areas for "Comments" (Machinery) and "Inspection".
- Dynamic Help Contents:** A pane showing the content of the selected "Comments" topic. It explains that the comments text areas are for reading and editing additional notes.

At the bottom left of the window, the copyright notice "© 2002-2004 JGoodies" is visible.

**Name**

**Surname**

**Search**

Nickname	Name	Surname	Phone
Vincent	Andrew	Gos	+1 400 789 00 78
Kitty	Kate	Phillie	+1 421 567 89 09

2009 - Мамонов Дмитрий - [42]

# Пример использования FormLayout:

```
public JComponent buildPanel() {
    initComponents();
    FormLayout layout = new FormLayout(
        "right:[40dlu,pref], 3dlu, 70dlu, 7dlu, "
        + "right:[40dlu,pref], 3dlu, 70dlu",
        "p, 3dlu, p, 3dlu, p, 3dlu, p, 9dlu, " +
        "p, 3dlu, p, 3dlu, p, 3dlu, p, 9dlu, " +
        "p, 3dlu, p, 3dlu, p, 3dlu, p");

    JPanel panel = new JPanel(layout);
    panel.setBorder(Borders.DIALOG_BORDER);
    // Fill the table with labels and components.
    CellConstraints cc = new CellConstraints();
    panel.add(createSeparator("Manufacturer"), cc.xyw(1, 1, 7));
    panel.add(new JLabel("Company:"), cc.xy(1, 3));
    panel.add(companyNameField, cc.xyw(3, 3, 5));
    panel.add(new JLabel("Contact:"), cc.xy(1, 5));
    panel.add(contactPersonField, cc.xyw(3, 5, 5));
    panel.add(new JLabel("Order No:"), cc.xy(1, 7));
    panel.add(orderNoField, cc.xy(3, 7));



    panel.add(createSeparator("Inspector"), cc.xyw(1, 9, 7));
    panel.add(new JLabel("Name:"), cc.xy(1, 11));
    panel.add(inspectorField, cc.xyw(3, 11, 5));
    panel.add(new JLabel("Reference No:"), cc.xy(1, 13));
    panel.add(referenceNoField, cc.xy(3, 13));
    panel.add(new JLabel("Status:"), cc.xy(1, 15));
    panel.add(approvalStatusComboBox, cc.xy(3, 15));

    panel.add(createSeparator("Ship"), cc.xyw(1, 17, 7));
    panel.add(new JLabel("Shipyard:"), cc.xy(1, 19));
    panel.add(shipYardField, cc.xyw(3, 19, 5));
    panel.add(new JLabel("Register No:"), cc.xy(1, 21));
    panel.add(registerNoField, cc.xy(3, 21));
    panel.add(new JLabel("Hull No:"), cc.xy(5, 21));
    panel.add(hullNumbersField, cc.xy(7, 21));
    panel.add(new JLabel("Project type:"), cc.xy(1, 23));
    panel.add(projectTypeComboBox, cc.xy(3, 23));

    return panel;
}
```



# Создание композитных элементов интерфейса

- Наследование от JPanel 
- Инкапсуляция компонента 



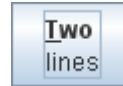
# Декорирование компонент

- <http://java.sun.com/docs/books/tutorial/uiswing/components/html.html>
- <http://java.sun.com/docs/books/tutorial/uiswing/components/border.html>

Для отображения форматированного текста в метках или на кнопках Swing поддерживает HTML (конечно не полный стандарт). Можно задавать стиль начертания символов, размер, цвет, создавать многострочные метки и даже использовать списки. Для того что бы в метке вместо текста был отображён HTML, строковое значени следуя задать с префиксом "<html>".

## Пример задания HTML содержимого компонента:

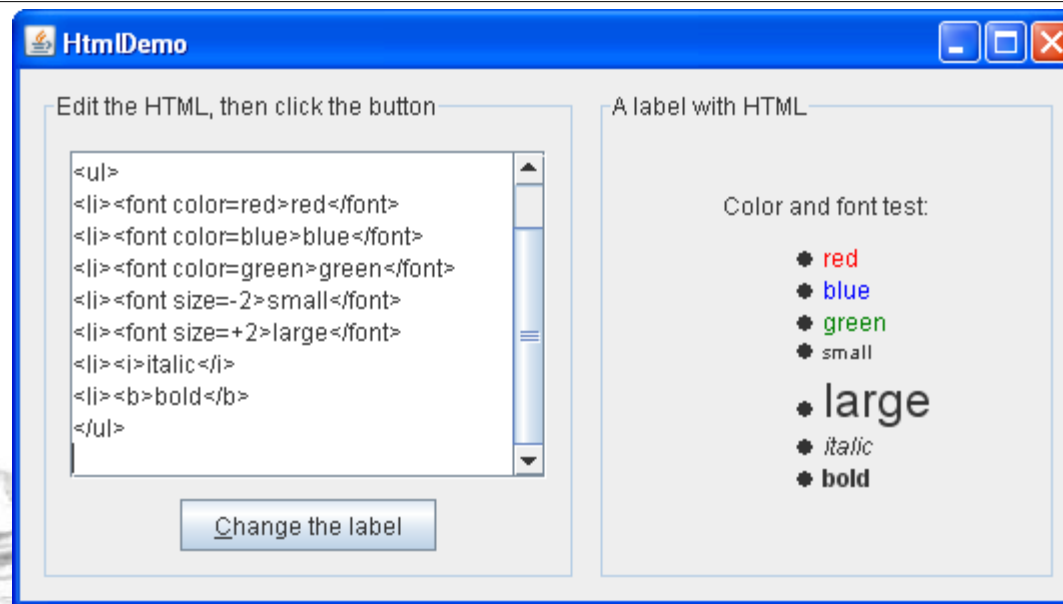
```
button = new JButton("<html><b><u>T</u>wo</b><br>lines</html>");
```



```
label = new JLabel("<html>\n" +  
    "Color and font test:\n" +  
    "<ul>\n" +  
    "<li><font color=red>red</font>\n" +  
    "<li><font color=blue>blue</font>\n" +  
    "<li><font color=green>green</font>\n" +  
    "<li><font size=-2>small</font>\n" +  
    "<li><font size=+2>large</font>\n" +  
    "<li><i>italic</i>\n" +  
    "<li><b>bold</b>\n" +  
    "</ul>\n");
```



# Пример использования HTML в метках:

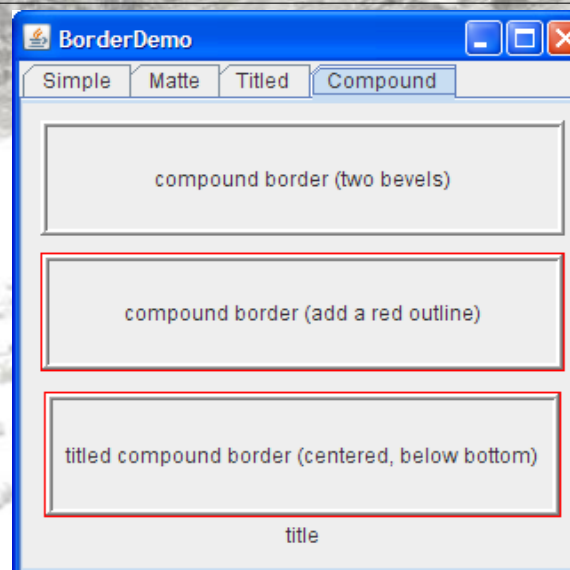
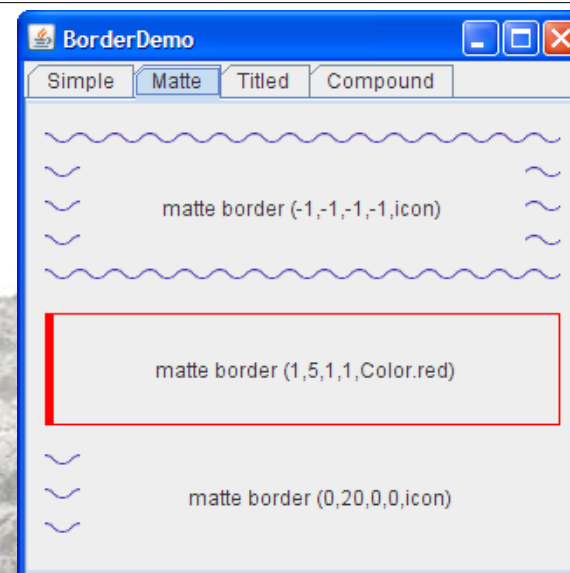
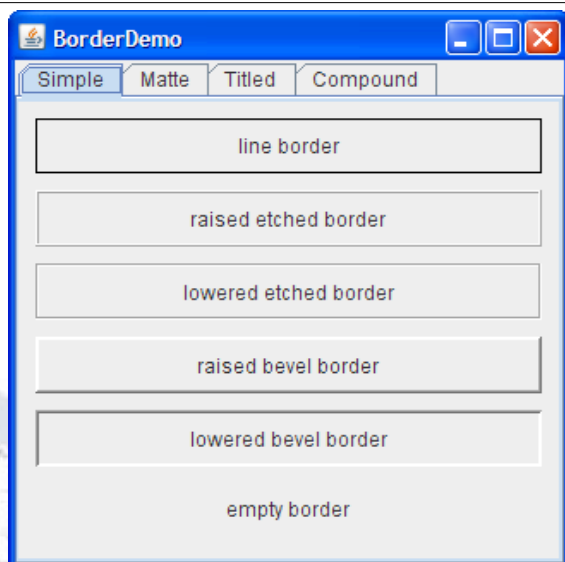


Для меток и кнопок можно задавать иконки и положение текста относительно иконок:



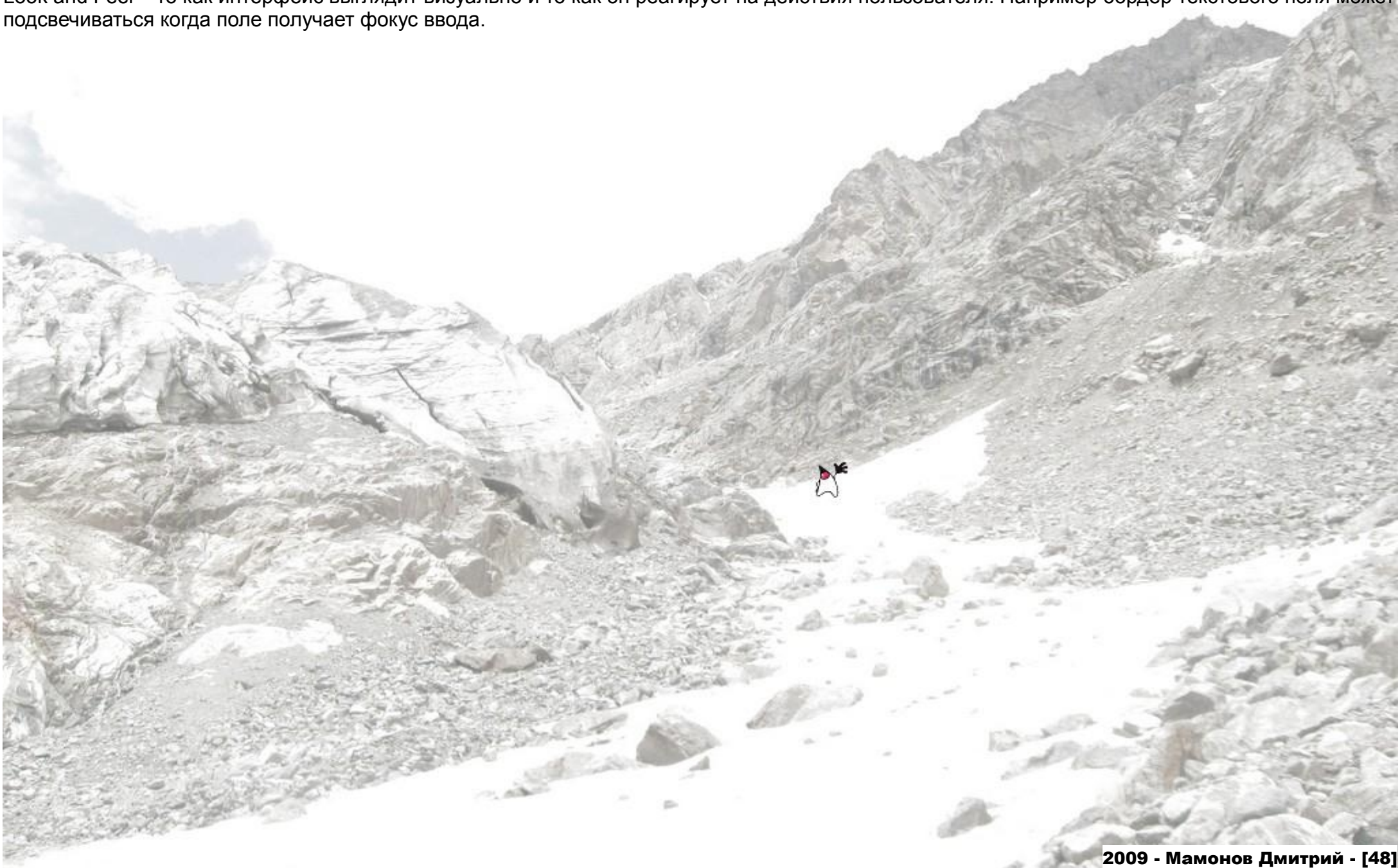


# Пример использования бордеров



# LaF: Nimbus, Looks, Substance

Look and Feel – то как интерфейс выглядит визуально и то как он реагирует на действия пользователя. Например бордер текстового поля может подсвечиваться когда поле получает фокус ввода.





# JGoodies Looks LaF

<http://www.jgoodies.com/freeware/looks/index.html>

<http://www.jgoodies.com/> (страница Karsten Lentzsch )

**Skeleton**

File Components Help

Navigator

- General Project Data
- Propeller Shaft1
  - Segment 1
  - Flange 1a
  - Flange 1b
- Intermediate Shaft
  - Segment 2a
  - Segment 2b
  - Flange 2a
  - Flange 2a
- Propeller Shaft2
  - Segment 3a
  - Segment 3b
  - Segment 3c
  - Flange 3a

**General Project Data**

**Project**

Identifier: Sample Project

**Manufacturer**

Company: Hapag Lloyd

Contact: Buzz Lightyear

Order No: 583-992/2002

**Inspector**

Name: Clouseau

Reference No: 32098

Status: In Progress

**Ship**

Shipyard: HDW

Register No: 22067

Hull Numbers: 472

Project Type: New Building

© 2002-2004 JGoodies.com

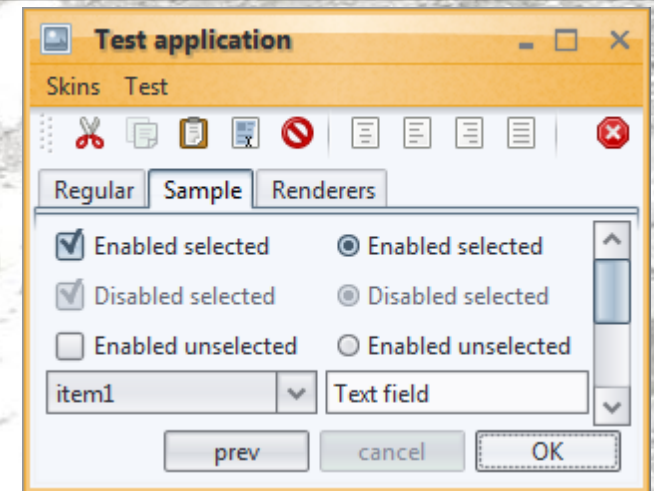
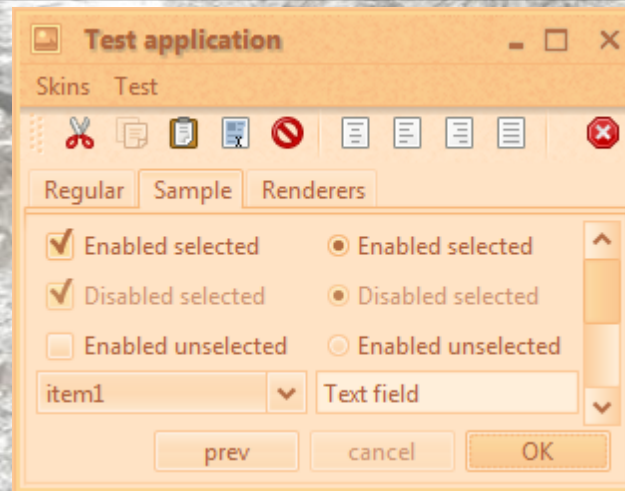
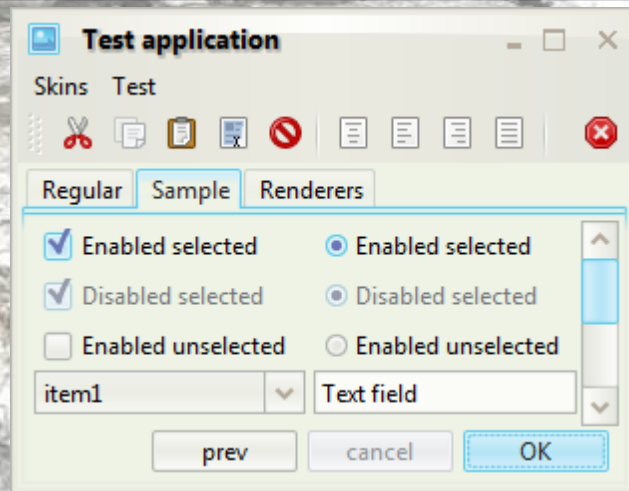
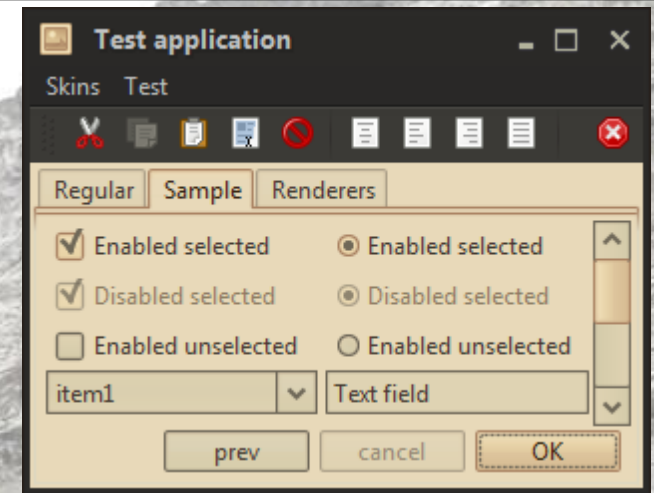
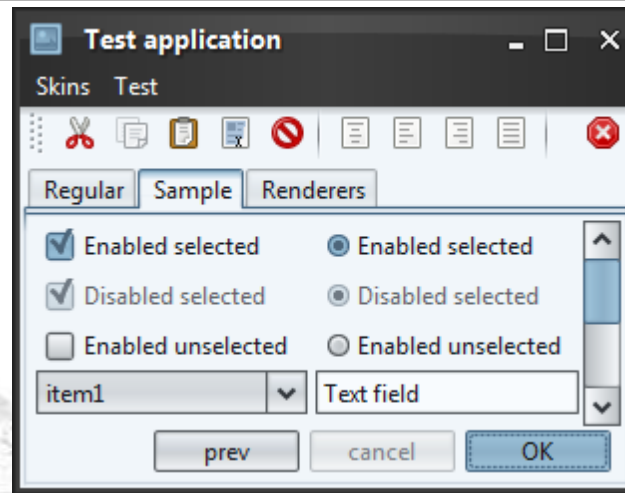
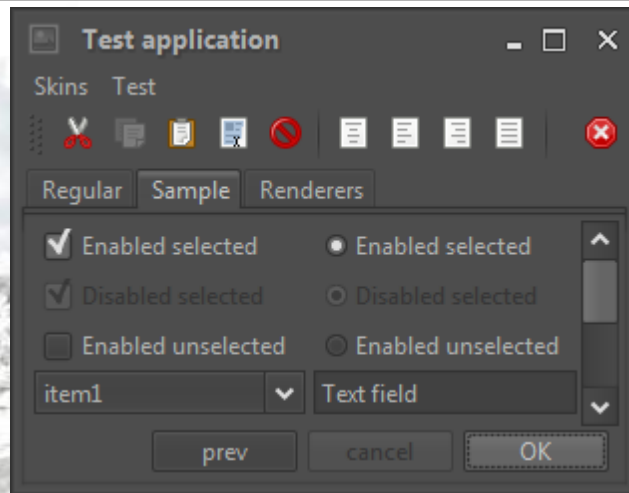


# Substance LaF

<https://substance.dev.java.net/see.html>

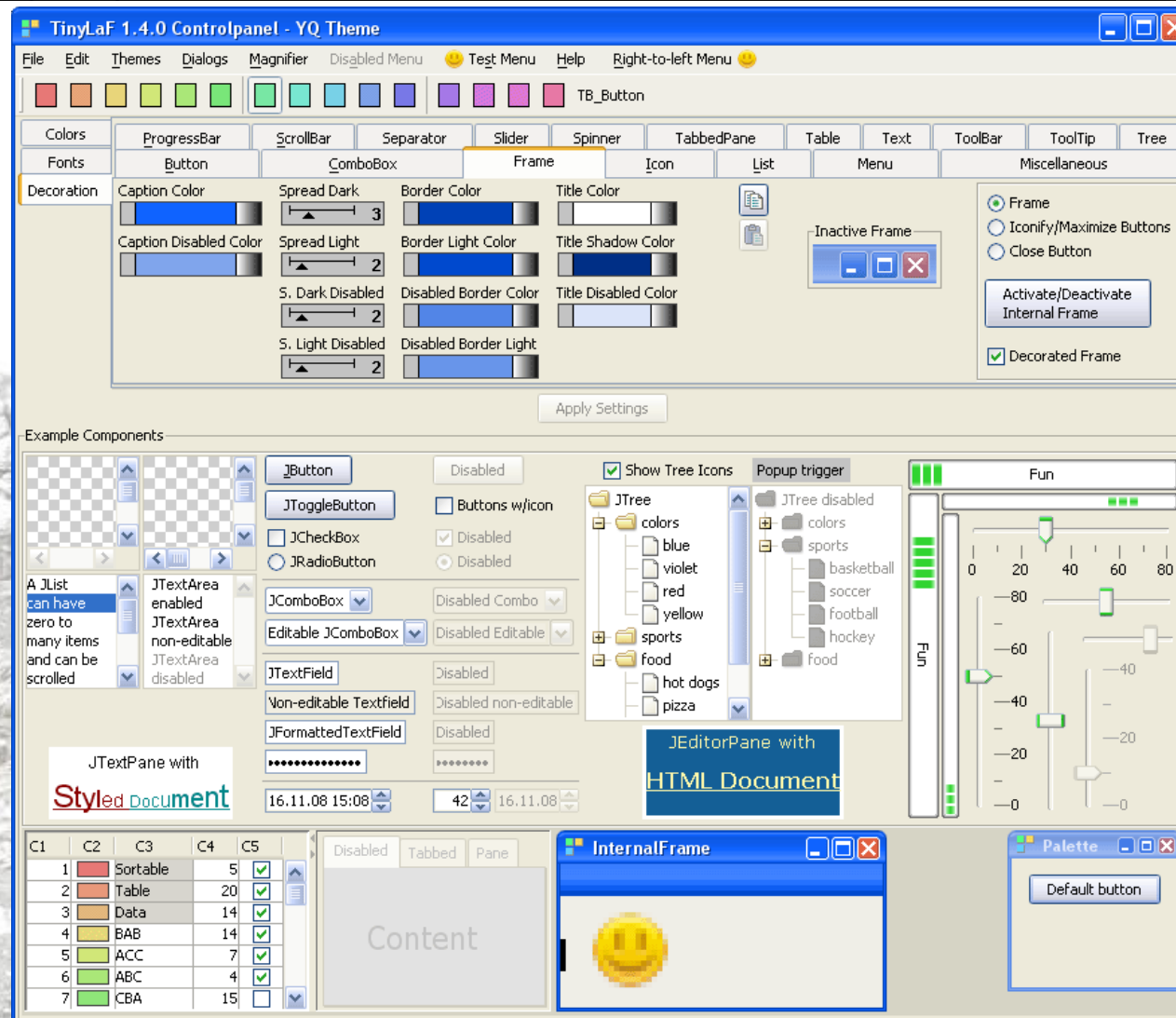
[http://www.pushing-pixels.org/?page\\_id=9](http://www.pushing-pixels.org/?page_id=9) (страница Kirill Grouchnikov)

Substance - яркий, поддерживает цветовые схемы



# TinyLaF

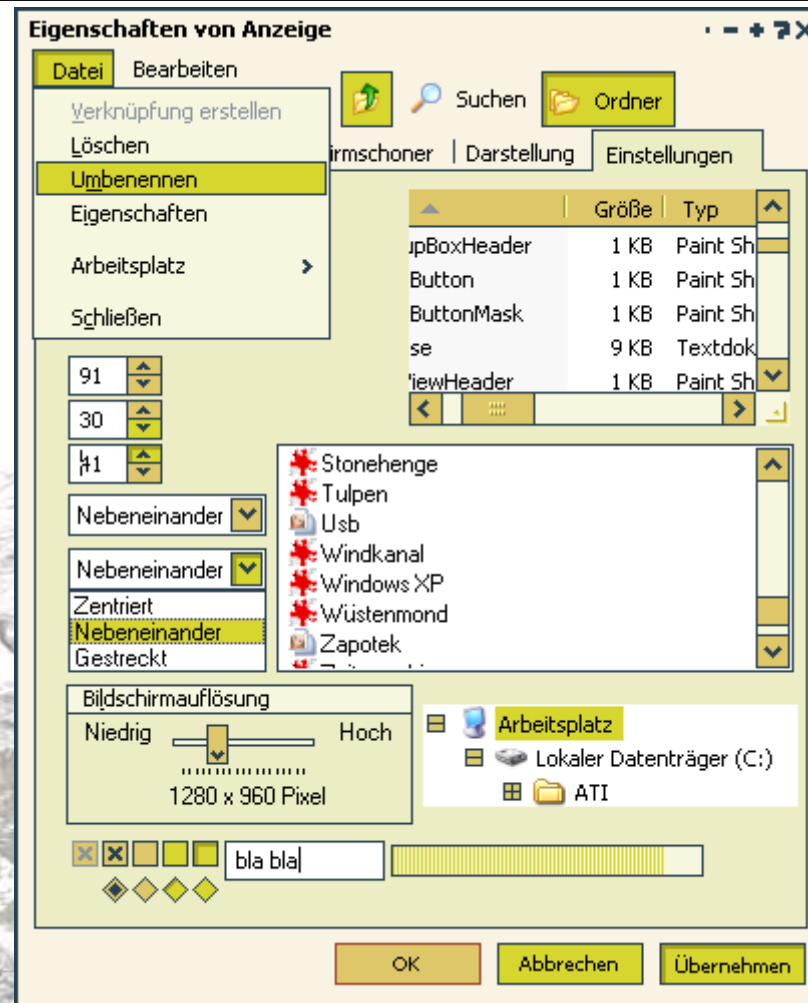
<http://www.muntjak.de/hans/java/tinylaf/tinyscreen.html>





# Squareness LaF

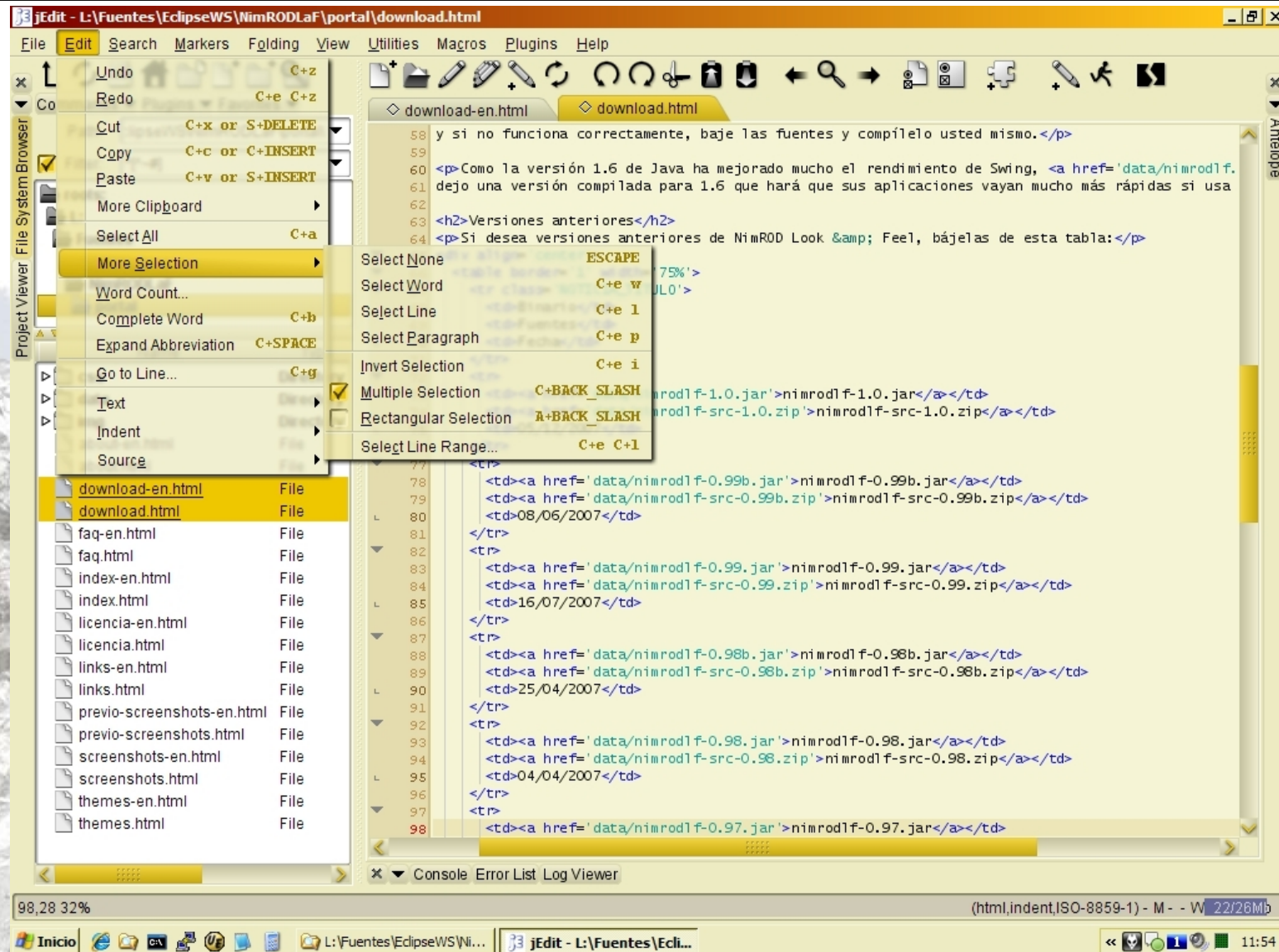
<http://squareness.beegeer.net/details.html>





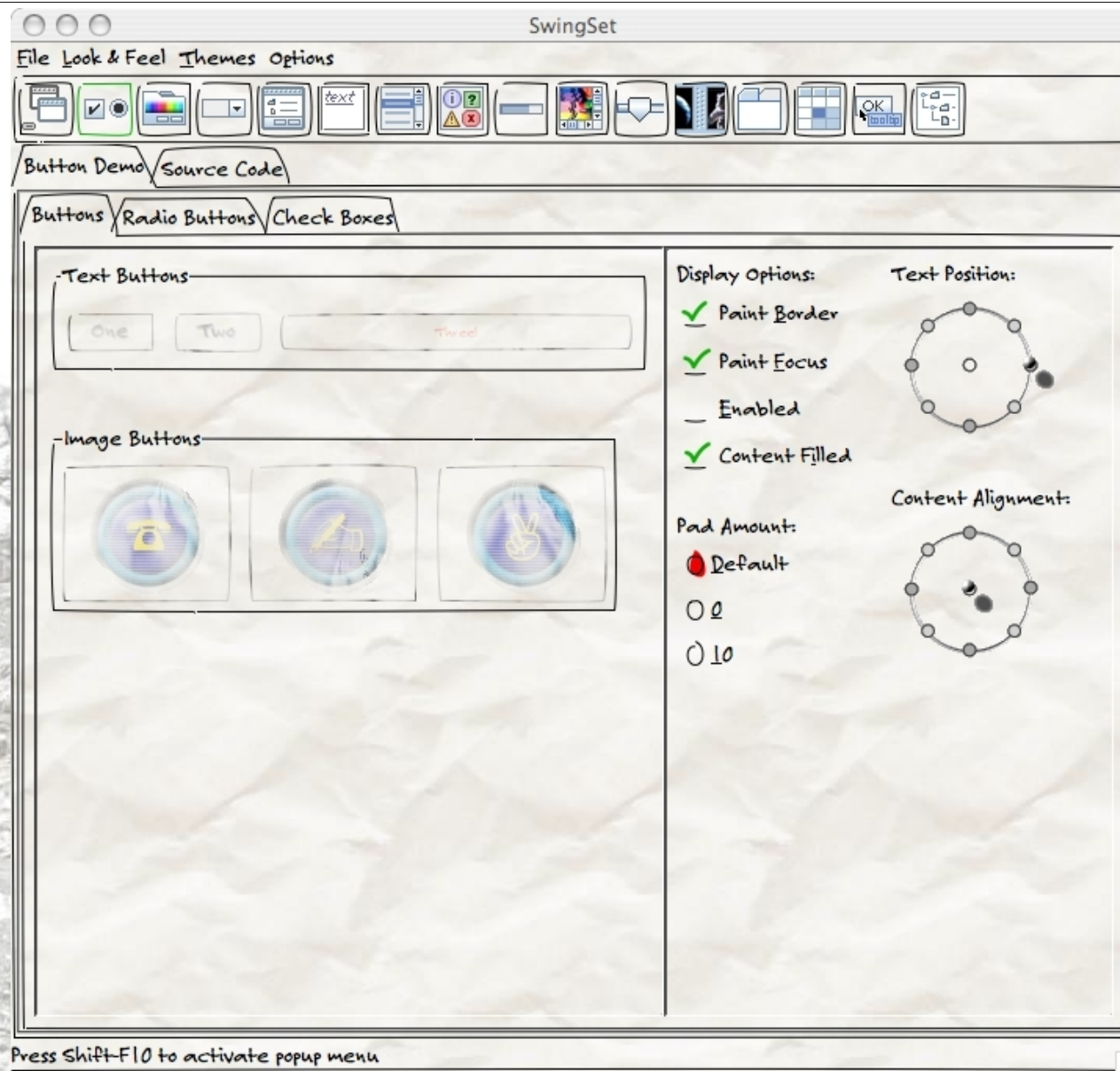
# Nimrod LaF

<http://personales.ya.com/nimrod/screenshots-en.html>



# Napkin LaF

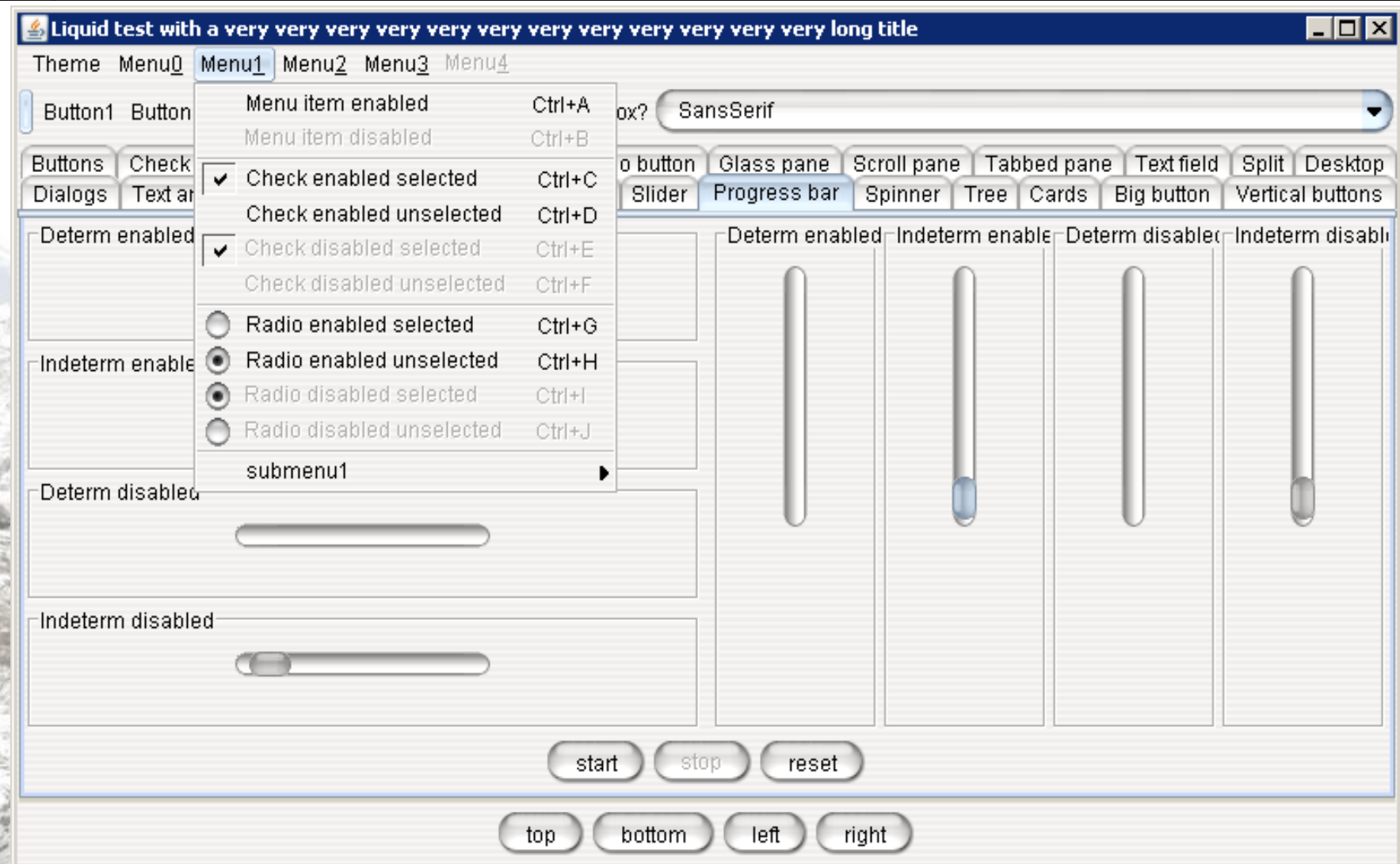
<http://napkinlaf.sourceforge.net/>





# Liquid LaF

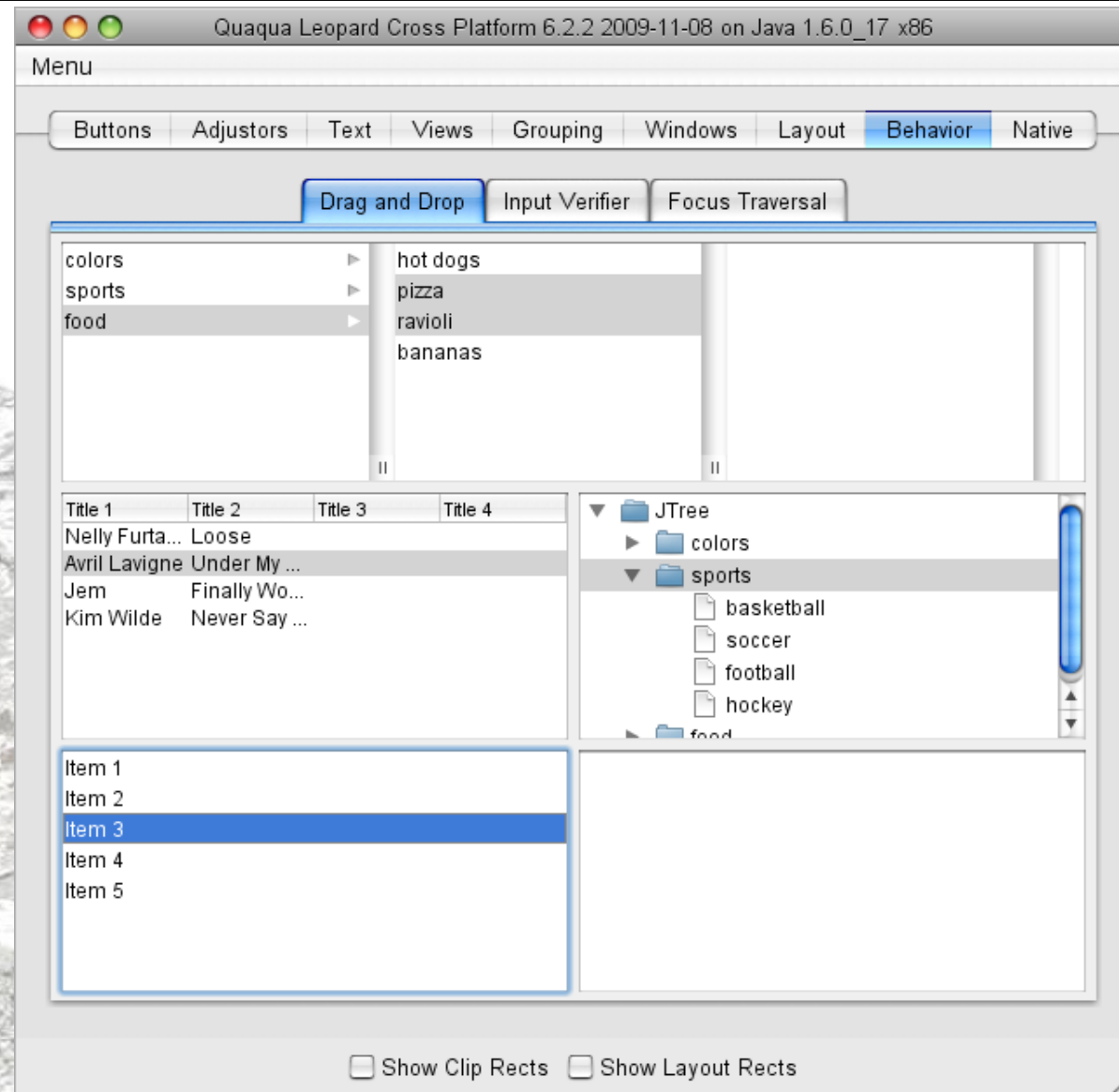
<https://liquidlnf.dev.java.net/>





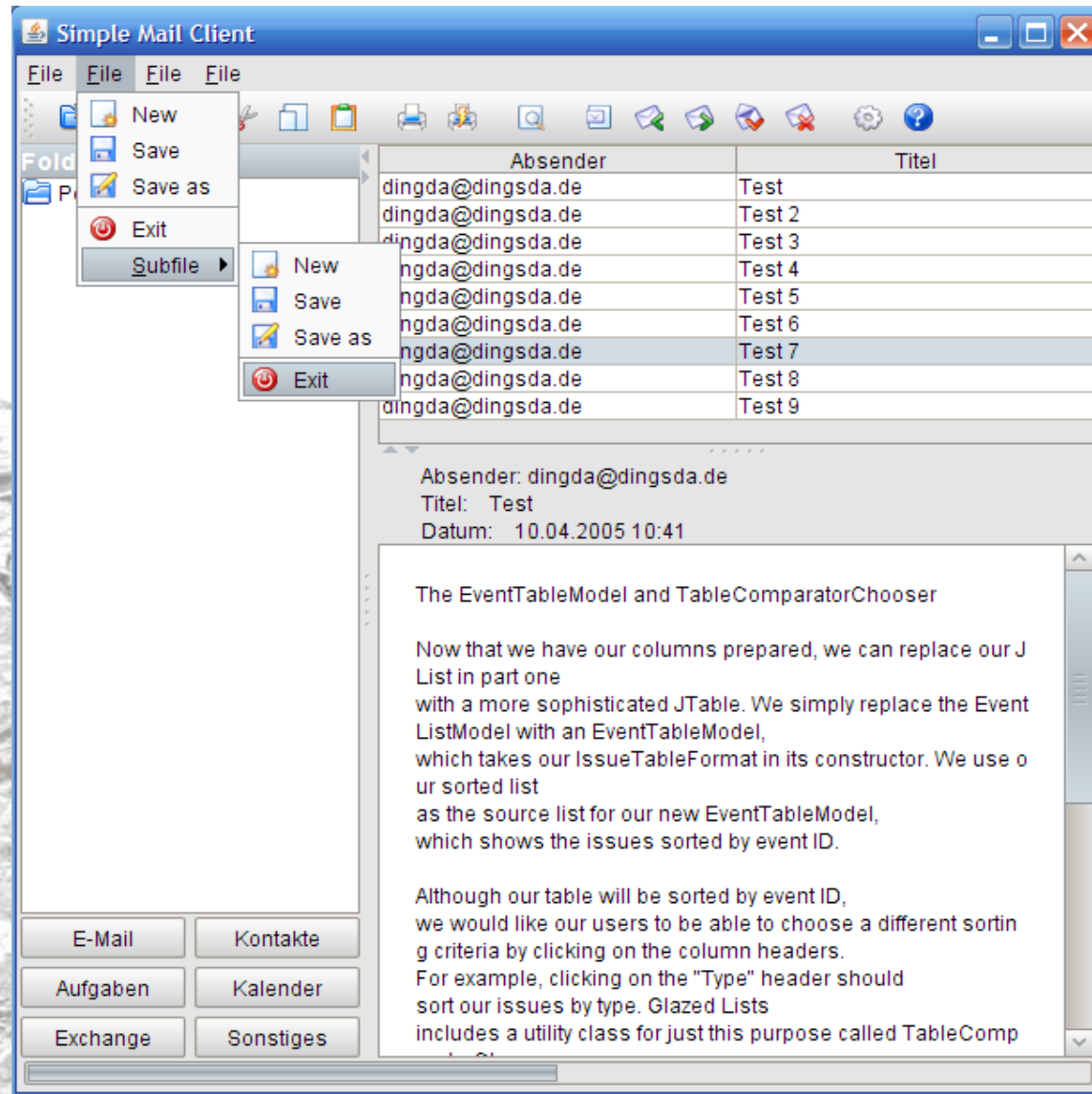
# QuaQua LaF

<http://www.randelshofer.ch/quaqua/index.html>



# PgsLookAndFeel

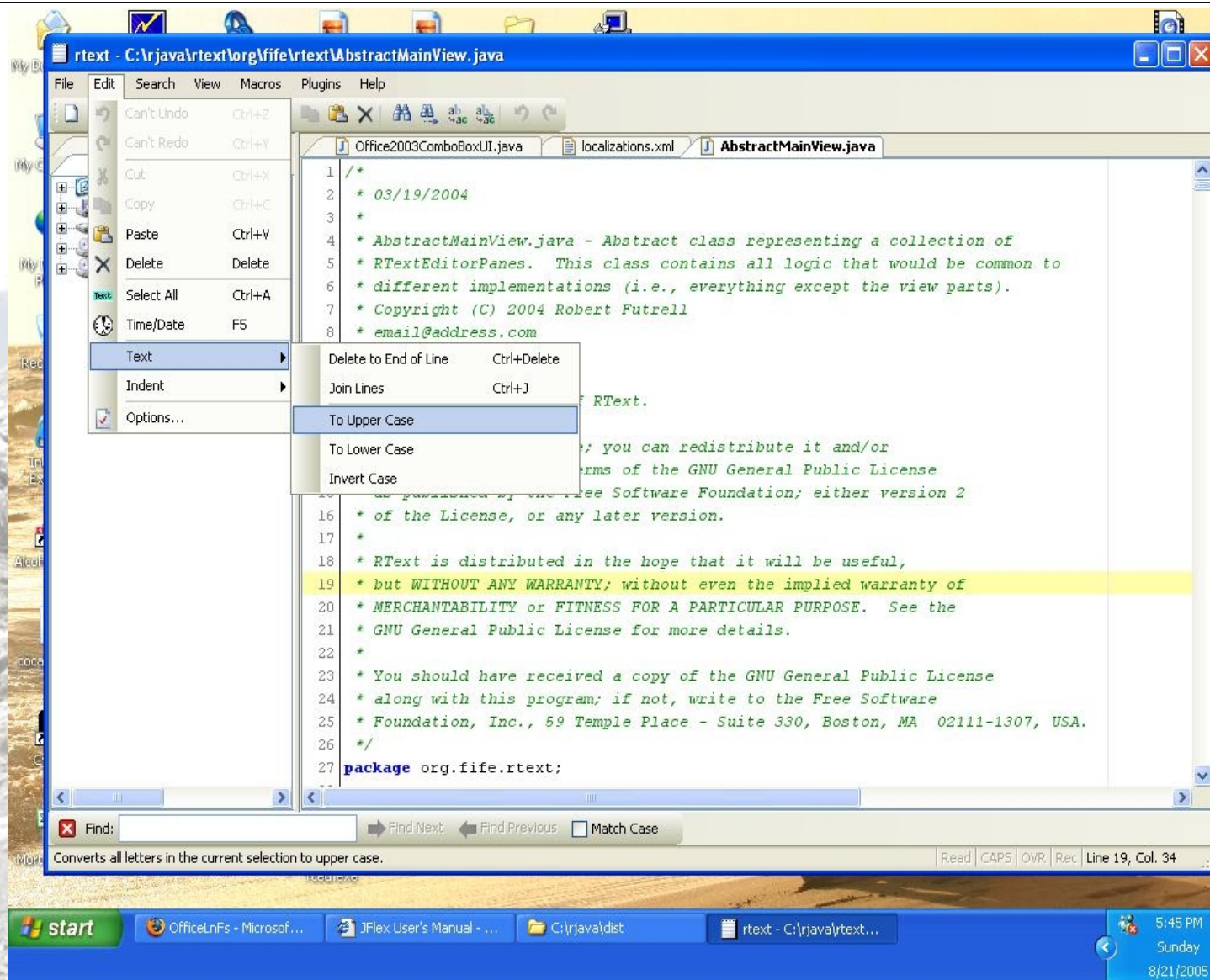
<https://pgslookandfeel.dev.java.net/>





# Office (MS) LaF

<http://fifesoft.com/officeInfs/>





# Nimbus LaF (включён в Sun Java начиная с 1.6.0\_u10)

<https://nimbus.dev.java.net/>

[http://java.sun.com/javase/6/docs/technotes/guides/jweb/otherFeatures/nimbus\\_laf.html](http://java.sun.com/javase/6/docs/technotes/guides/jweb/otherFeatures/nimbus_laf.html)

<http://java.sun.com/developer/technicalArticles/javase/java6u10/>

★ Морена - поддержка соц. исследований по методу Морено

Исследование Навигация Утилиты Настройки Помощь

Назад Далее

**Шаги исследования**

1. Создать новое исследование
2. Задать параметры исследования
3. Задать результаты опроса
4. Анализ: социоматрица
5. Анализ: социограмма
6. Заключение

Назад | Далее

**Задание параметров исследования**

Социометрическое исследование начинается с определения задачи исследования.

В поле [описание исследования](#) опишите рассматриваемую вами целевую группу, цели исследования и факторы ему сопутствующие.

Пример: Изучение деловых отношений в коллективе персонала кафе "Чайный домик". Цель: определить качество отношений в коллективе.

Укажите информацию о себе в поле [Психолог-Исследователь](#). По умолчанию используется имя текущего пользователя в операционной системе.

Done : actionNavigateNext

**Описание исследования**

Название исследования: Пример: Исследование группы школьников...

Цель исследования:

Исследуемая группа: Пример: Ученики 7б класса, школа 12, 2009г.

Психолог: Фамилия И.О. Дата опроса: дд.мм.гггг чч:мм

Тип опроса: ☐ Непараметрический опрос ☒ Параметрический вопрос

**Опросник**

Как отвечать на вопросы: Пример: Введение в опросник для респондента...

+ Добавить вопрос Печать

**Вопрос 1**

Тип вопроса: Предпочтение Метка: + Вверх Вниз Удалить

Позитивный вопрос

Дамми

# Установка Look and Feel:

Для совместимости Metal LaF оставлен как представления по умолчанию. Но сменить LaF на другой легко:

```
try {
    for (LookAndFeelInfo info : UIManager.getInstalledLookAndFeels()) {
        if ("Nimbus".equals(info.getName())) {
            UIManager.setLookAndFeel(info.getClassName());
            break;
        }
    }
} catch (UnsupportedLookAndFeelException e) {
    // handle exception
} catch (ClassNotFoundException e) {
    // handle exception
} catch (InstantiationException e) {
    // handle exception
} catch (IllegalAccessException e) {
    // handle exception
}
```

Важно, этот код должен быть выполнен до инициализации первого визуального компонента, в противном случае тот может отобразиться с LaF по умолчанию. Возможные проблемы:

**Problem:** My application is not showing the look and feel I have requested via `UIManager.setLookAndFeel`.

You probably either set the look and feel to an invalid look and feel or set it after the UI manager loaded the default look and feel. If you are sure that the look and feel you specified is valid and setting the look and feel is the first thing your program does (at the top of its main method, for example), check whether you have a static field that references a Swing class. This reference can cause the default look and feel to be loaded if none has been specified. For more information, including how to set a look and feel after the GUI has been created, see the [look and feel](#) section.

<http://java.sun.com/docs/books/tutorial/uiswing/misc/problems.html>

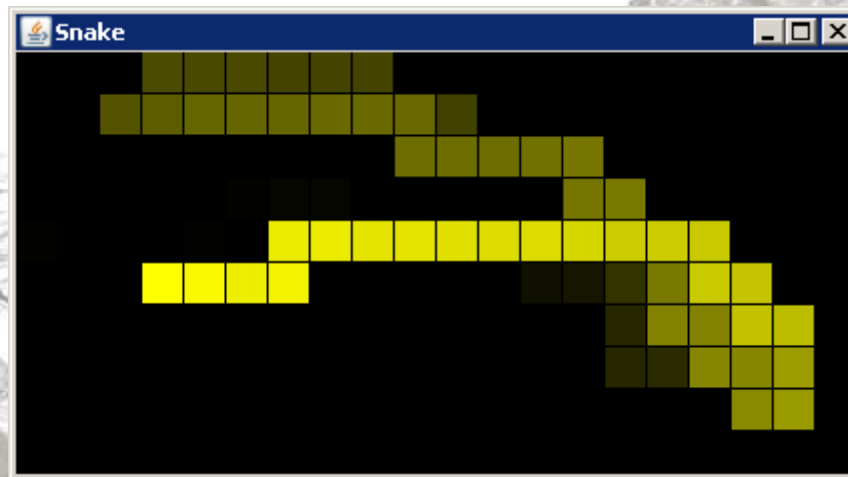
## Так же LaF для приложения можно установить через параметры виртуальной машины:

```
java -Dswing.defaultlaf=com.sun.java.swing.plaf.nimbus.NimbusLookAndFeel ...
```

# Создаие анимации

## Базовые принципы

- <http://kenai.com/projects/trident/pages/Home>
- <http://incubator.apache.org/pivot/1.4/tutorials/effects.transitions.html> (Pivot)
- <http://www.jgoodies.com/downloads/libraries.html>

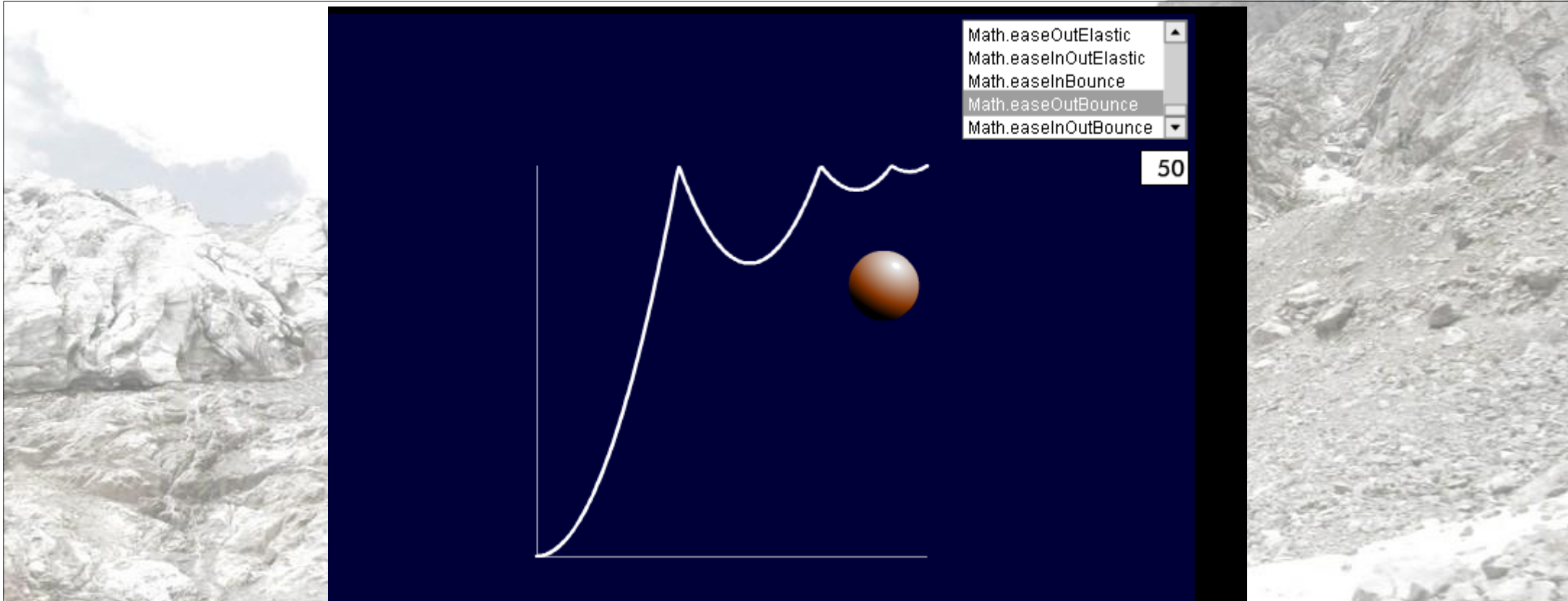




# Эффекты анимации Robert-a Penner-a

- <http://robertpenner.com/>
- [http://robertpenner.com/easing/easing\\_demo.html](http://robertpenner.com/easing/easing_demo.html)

Easing - Интересные способы изменения параметра анимации со страницы Robert Penner-a.



# Интернационализация приложения☰

- Поддержка приложением нескольких языков интерфейса
- Выбор языка подходящего для пользователя
- Инкапсуляция значений зависящих от языка

```
JLabel label = new  
JLabel(I18N.get("question_editor.question_type_label"));
```

## ResourceBundle — Встроенная поддержка

Один из вариантов — <http://java.sun.com/javase/6/docs/api/java/util/ResourceBundle.html>.

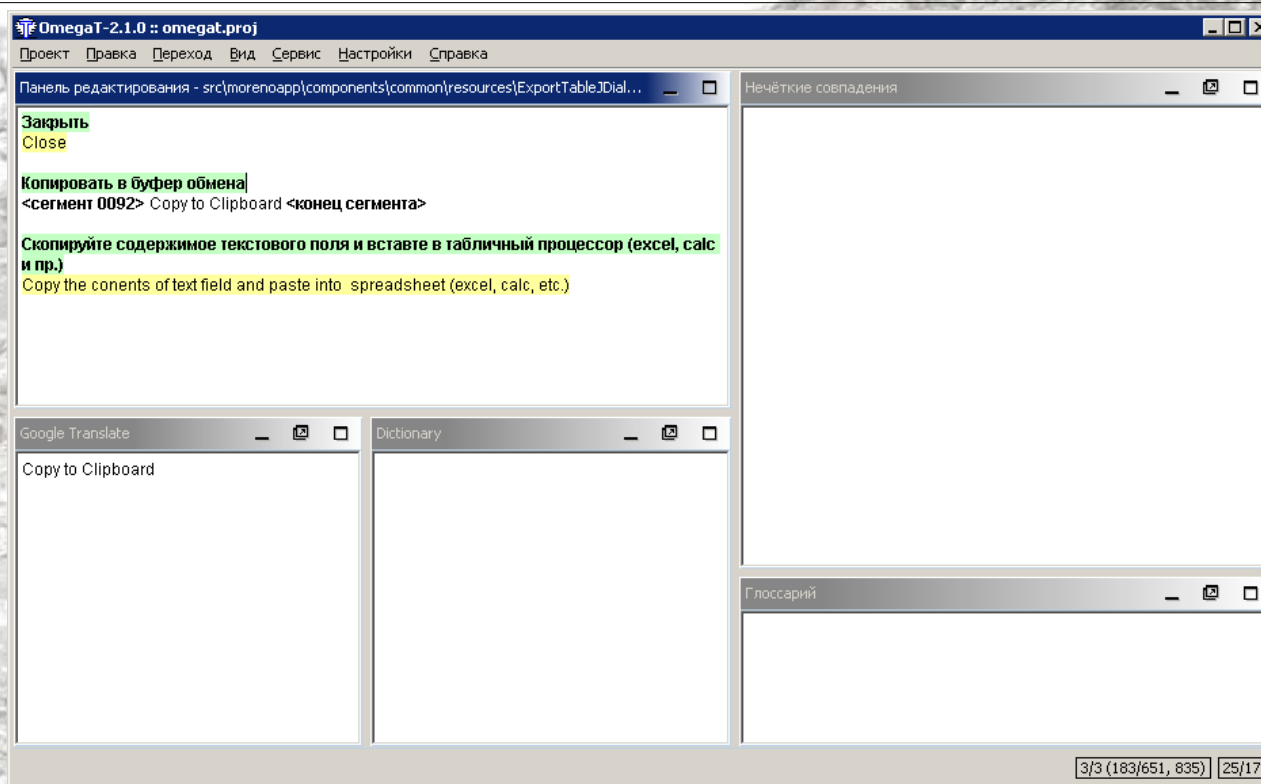
## SwingAppFramework

Поддерживает интернационализацию приложения через ResourceMap (который в свою очередь использует ResourceBundle).



# OmegaT — Унификация и перевод

- <http://www.omegat.org/>
- Работает напрямую с кодом проекта
- Подсказка при переводе — Google Translate
- Похожие переводы — поддержка терминологии

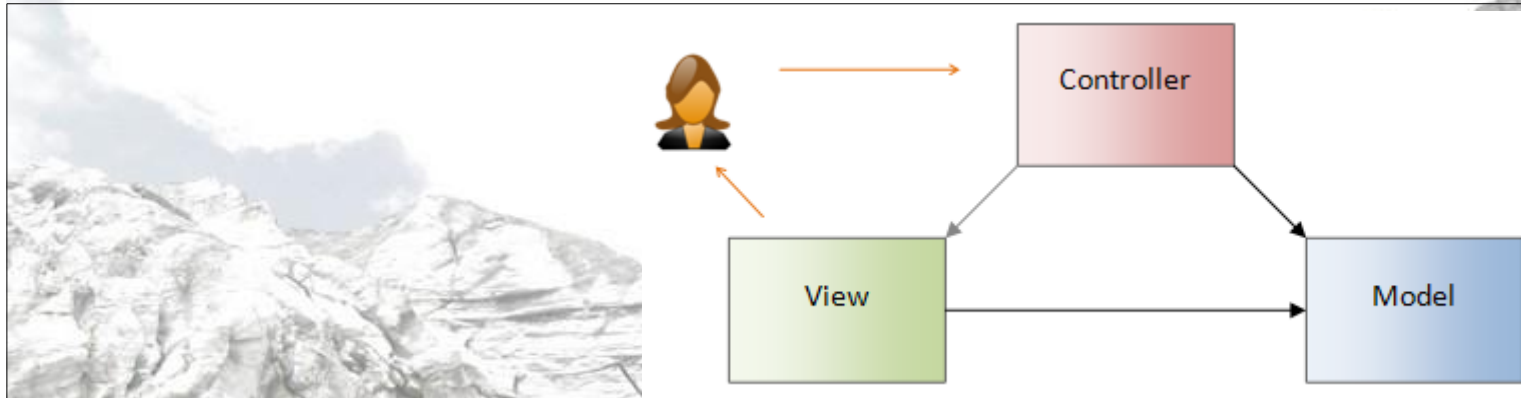




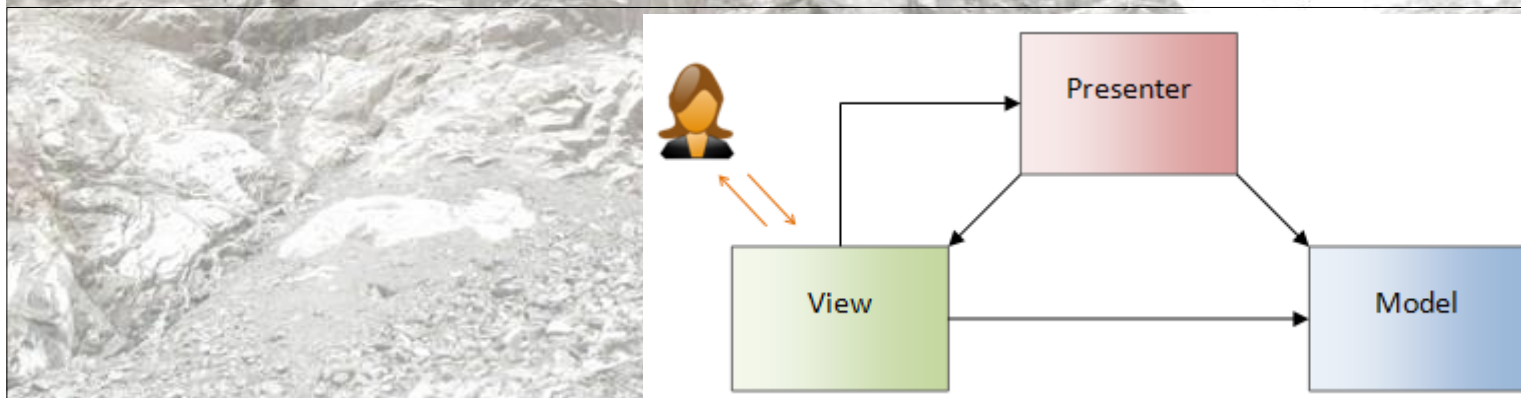
# Модель данных приложения

- <http://wiki.apidesign.org/wiki/MVC>, <http://wiki.apidesign.org/wiki/DCI>, <http://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller>
- <http://java.sun.com/products/jfc/tsc/articles/architecture/>, <http://t-deli.com/listeners.html>

## Model View Controller

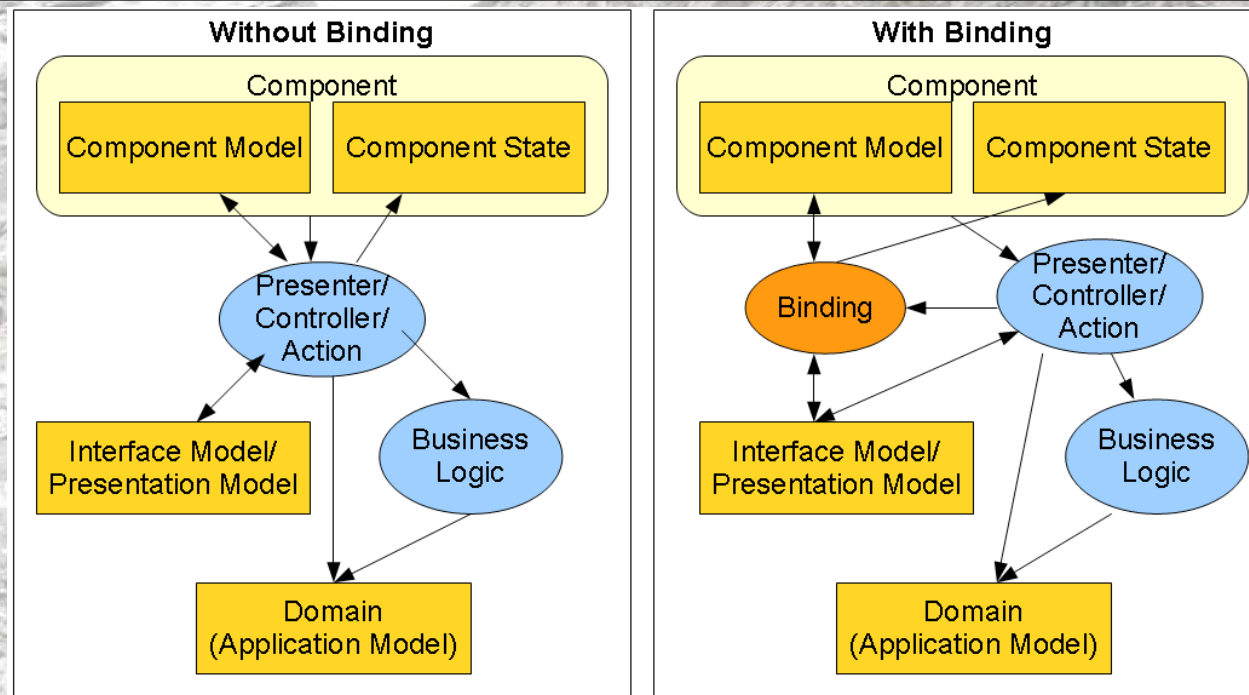


## Model View Prosentation



# Структура Swing приложения

- Модель данных компонента - TableModel
- Состояние компонента – isEnabled(), isVisible()
- Контроллер - ActionListener
- Модель данных интерфейса - OrderFormBean
- Бизнес логика – CRUD for ORDER
- Модель данных приложения – PersistentOrder/TABLE ORDER
- Binding – часть контроллера которая связывает атрибуты модель компонента и атрибуты модель интерфейса.





# Binding – just don't do it (or Delphi approach)

Цель связывания модели данных интерфейса и модели приложения в их разделении. Связанные модели ведут себя синхронно, но являются разными сущностями, а значит их можно использовать независимо. Например, реализовать логику пакетной обработки модели которая вообще не взаимодействует с интерфейсом.

Но всегда ли это нужно?

Если приложение является тонким клиентом работающим в режиме запрос/ответ, и взаимодействующим с сервером передавая JSON данные по http, нужен ли здесь биндинг.

- Запрос
  1. Поместить данные компонента в бин
  2. поместить данные бина в JSON
  3. преобразовать JSON в строку и отправить запрос
- Ответ
  1. Преобразовать полученную строку в JSON
  2. разобрать JSON и построить по нему бин
  3. Обновить компонент по значению в бине

Если логика формы простая — ограниченный область данных для работы, простая валидация (на уровне 1 поля) то 2-й шаг можно опустить, данные просто копируются в бин и обратно, при этом никакой логики не происходит. 🐛

Такой подход используется в pivot, хотя pivot имеет одно преимущество. Модель данных построена так что Map/Bean/JSON это один и тот же интерфейс, по этому в действительности мы всё ещё может продолжать работат со структурированными объектами (бинами) хотя реально они будут оспользоваться только для преобразования данных в другой формат.

В Delphi, если логика реализована на хранимых процедурах, всё что должно сделать приложение — вызвать процедуру. Реальная модель данных приложения — набор параметров хранимой процедуры, и дублировать её ещё раз возможно дествительно смысла не имеет.

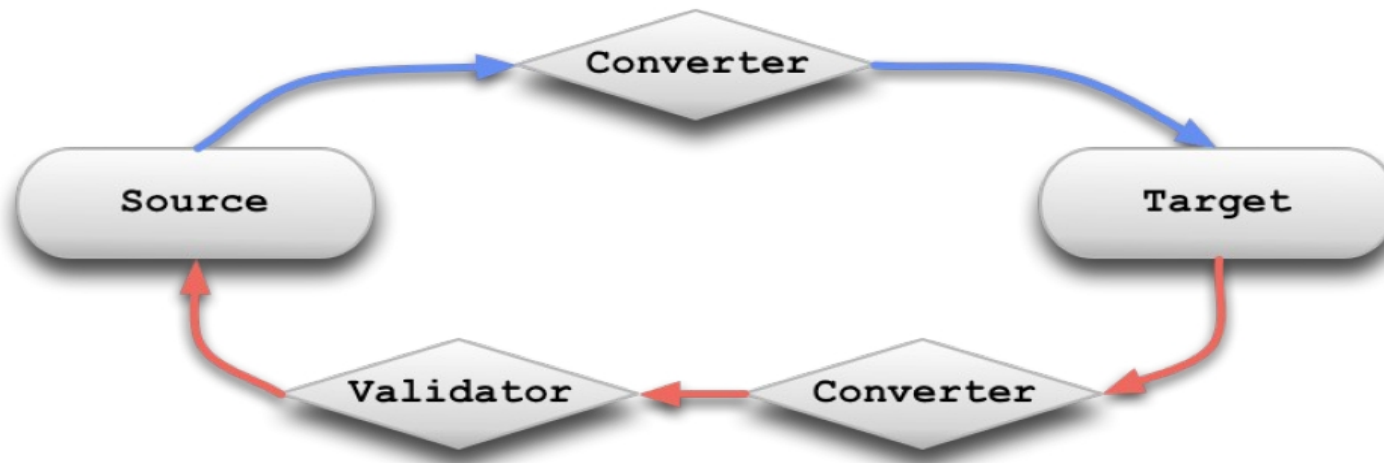
JavaScript приложения так же не имеют отдельной модели данных.



# Beansbinding

- <https://beansbinding.dev.java.net/>
- <http://developers.sun.com/learning/javaoneonline/2007/pdf/TS-3569.pdf>
- <http://blog.marcnuri.com/blog/default/2008/11/18/BeansBinding-Performance-Issue-37>
- <http://swinglabs.org/docs/presentations/2007/DesktopMatters/beans-binding-talk.pdf>
- <http://java.sun.com/javase/6/docs/api/java/beans/PropertyChangeListener.html>
- Интегрирован с NetBeans.
- Строится на связывании атрибутов бинов, не обязательно что бы они были визуальными.
- Обзор изменений осуществляется через PropertyChangeListener, бин должен быть Observable - реализовывать методы:
  - addPropertyChangeListener
  - removePropertyChangeListener
- Если один или оба из связанных бинов не реализуют эти методы, то они будут доступны только для чтения/записи. Но изменения в самом бине в этом случае не будут пропагироваться.
- Для связывания атрибутов различных типов используются конвертеры — интерфейс с парой методов для преобразования из исходного типа в целевой и обратно.
- Связанные атрибуты можно объединять в группы, это даёт возможность включать/выключать механизм биндинга одновременно для нескольких связей.
- BeansBinding поддерживает отображение списка на модель данных таблицы, или списка. Но реально это не до конца реализовано.
- Кроме того, библиотека имеет известные проблемы с производительностью и по всей видимости разработка заморожена с 2007 года.

# Beansbinding в действии



```
BindingConver colorConverter = new BindingConverter() {  
    public void Object sourceToTarget(Object value){  
        if (((Boolean)value).booleanValue()) return Color.GREEN;  
        else return Color.RED;  
    }  
};
```

```
CustomerBean customer = ...  
JTextField customerName = ...
```

```
Binding binding = new Binding(/*src*/customer,"${active}", /*target*/textField, "background");  
binding.setConverter(colorConverter);  
binding.bind();
```

# JGoodies binding

- <http://www.jgoodies.com/articles/>
- <http://www.jgoodies.com/articles/binding.pdf>
- <http://www.jgoodies.com/articles/patterns-and-binding.pdf>
- <http://www.jgoodies.com/downloads/libraries.html>

```
HintTextField text = new HintTextField();
text.setName("textResearchName");
text.setHint(I18N.get("config.research_name_hint"));
//binding
ValueModel value = new PropertyAdapter(model, "name", true);
Bindings.bind(text, value);
```





# GlazedLists

- <http://www.publicobject.com/glazedlists/>

```
.....
public class IssuesBrowser {
    /** event list that hosts the issues */
    private EventList<Issue> issuesEventList = new BasicEventList<Issue>();

    /** Create an IssueBrowser for the specified issues. */
    public IssuesBrowser(Collection<Issue> issues) {
        issuesEventList.addAll(issues);
    }

    /** Display a frame for browsing issues. */
    public void display() {
        // create a panel with a table
        JPanel panel = new JPanel();
        panel.setLayout(new GridBagLayout());

        EventListModel<Issue> issuesListModel = new EventListModel<Issue>(issuesEventList);
        JList issuesJList = new JList(issuesListModel);

        JScrollPane issuesListScrollPane = new JScrollPane(issuesJList);
        panel.add(issuesListScrollPane, new GridBagConstraints(...));
        // create a frame with that panel
        .....
    }
    .....
    issuesEventList.add(issue); //and new issue will appear in JList!!!
    .....
}
```

# Валидация данных

- <https://www.hibernate.org/412.html> (Hibernate Validator)
- <https://www.hibernate.org/412.html>
- <http://www.jgoodies.com/articles/validation.pdf>
- <https://validation.dev.java.net/>
- <http://weblogs.java.net/blog/2007/11/02/beans-binding-converter-and-validator-samples>

```
public class Address {
    @NotNull private String line1;
    private String line2;
    private String zip;
    private String state;
    @Length(max = 20) @NotNull private String country;
    @Range(min = -2, max = 50, message = "Floor out of range") public int floor;
}

private void validate(Object _param) {
    ValidatorFactory vf = Validation.buildDefaultValidatorFactory();
    Validator validator = vf.getValidator();
    Set<ConstraintViolation<Object>> constraintViolations = validator.validate(_param);

    if (constraintViolations.size() > 0) {
        StringBuilder sbResult = new StringBuilder("Invalid arguments:\n");
        int index = 0;
        for (ConstraintViolation<Object> cv : constraintViolations) {
            index++;
            Object value = cv.getInvalidValue();
            sbResult.append(String.format(
                "%d. %s='%s' - %s\n",
                index, cv.getPropertyPath(), display, cv.getMessage()));
        }
        throw new IllegalArgumentException(sbResult.toString());
    }
}
```

# Выполнение ресурсоёмких вычислений

## Swing Concurrency Model

- <http://java.sun.com/docs/books/tutorial/uiswing/concurrency/index.html>
- [http://weblogs.java.net/blog/kgh/archive/2004/10/multithreaded\\_t.html](http://weblogs.java.net/blog/kgh/archive/2004/10/multithreaded_t.html)

Потоки Swing приложения: 

### 1. Инициализирующий поток

```
SwingUtilities.invokeLater(new Runnable() {  
    public void run() {  
        createAndShowGUI();  
    }  
});
```

### 2. Event Dispatch Thread

### 3. Worker thread(s)

```
final int inputA = 2; final int inputB = 2;  
  
new Thread(new Runnable() {  
    public void run() {  
        final int resut = inputA+inputB;  
        SwingUtilities.invokeLater(new Runnable() {  
            public void run() {  
                labelStatus.setText(inputA + " + " + inputB + " = " + result);  
            }  
        });  
    }  
}).start();
```



# SwingUtilities

SwingUtilities – вспомогательный класс для работы с Event Dispatch Thread и некоторых других задач. Основные методы:

- **invokeLater(Runnable)** – асинхронно выполняет переданный код в EDT, после вызова метода мы сразу получим управления назад. Переданный код будет помещён в очередь событий интерфейса и будет выполнен когда до него дойдёт очередь.
- **invokeAndWait(Runnable)** — выполняет переданный код в нити EDT синхронно. Нить вызывающая invokeAndWait будет заблокирована до тех пор, пока переданный код не будет выполнен.
- **isEventDispatchThread()** - проверяет, является ли текущая нить исполнения нитью EDT или другой нитью. Если один и тот же код может выполняться внутри EDT то все операции с интерфейсом можно производить безопасно, в противном случае следует передать код работающий с интерфейсом в EDT, например через invokeLater().

# SwingWorker Framework

- <https://swingworker.dev.java.net/>
- <https://swingworker.dev.java.net/nonav/javadoc/org/jdesktop/swingworker/SwingWorker.html>
- <http://java.sun.com/javase/6/docs/api/javax/swing/SwingWorker.html>
- <http://foxtrot.sourceforge.net/> - похожее решение, дизайн в духе AJAX (post/success/failure)

Функциональность SwingUtilities достаточная, но часто нужно сделать следующее:

1. Запустить длительную задачу на исполнение в фоновом процессе
2. Отображать прогресс выполнения операции в интерфейсе (найдено 5 файлов, 6...)
3. Возможность отменить задание в ходе выполнения
4. По окончании выполнения операции (либо ошибка) отобразить результат в интерфейсе.

Всё это есть в SwingWorker — он доступен в виде отдельной библиотеки, так же включён а Java 6.



# SwingWorker: Пример

## Основные функции:

- 1.Выполнение задач в фоновом потоке.
- 2.Уведомление о ходе выполнения через изменение Observable свойств.
- 3.Возможность принудительного завершения потока
- 4.Возможность публикации частичных результатов (publish() и process())
- 5.Параметры T — тип общего результата, V – частичный результат

```
final JLabel label;  
class MeaningOfLifeFinder extends SwingWorker<String, Object> {  
    @Override  
    public String doInBackground() {  
        return findTheMeaningOfLife();  
    }  
  
    @Override  
    protected void done() {  
        try {  
            label.setText(get());  
        } catch (Exception ignore) {  
        }  
    }  
}  
(new MeaningOfLifeFinder()).execute();
```



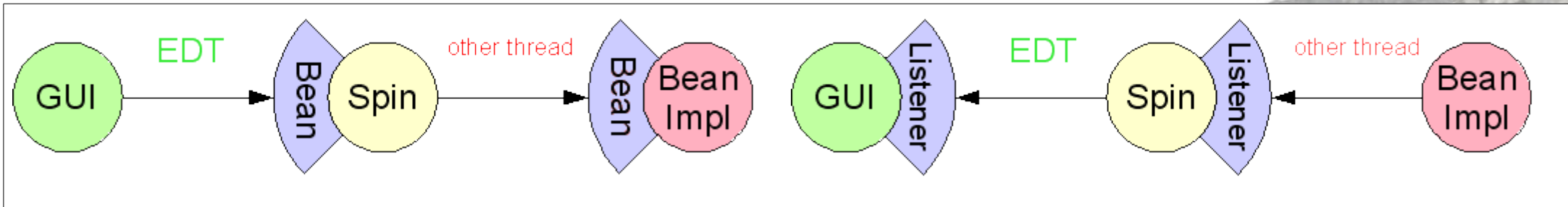
# javax.swing.Timer

- <http://java.sun.com/docs/books/tutorial/uiswing/misc/timer.html>
- <http://java.sun.com/javase/6/docs/api/javax/swing/Timer.html>

```
final javax.swing.Timer timer = new javax.swing.Timer(500, null);
ActionListener actionListener = new ActionListener() {
    int countDown = 10;
    @Override
    public void actionPerformed(ActionEvent event) {
        System.out.println("CountDown: "+countDown);
        countDown--;
        if (countDown<=0) {
            timer.stop();
        }
    }
};
timer.addActionListener(actionListener);
timer.start();
```

# Другие решения: spin

- <http://spin.sourceforge.net/>



*Spin* is built on top of virtual proxies and a technique borrowed from the `java.awt.Dialog` component. While a modal dialog has to wait for user input, the EDT is rerouted to the swing event processing to handle further events.

GUI.java

```
public void actionPerformed(ActionEvent e)
{
    label.setText("...");
    label.setText(bean.getValue());
}

public void propertyChange(PropertyChangeEvent ev)
{
    label.setText((String)ev.getNewValue());
}
```

BeanImpl.java

```
public String getValue()
{
    String value;
    // extensive calculation
    return value;
}

public void setValue(String value)
{
    this.value = value;
    firePropertyChange(value);
}
```

# Генерация отчётов и документов

## POI & OpenOffice

- <http://poi.apache.org/>
- <http://poi.apache.org/spreadsheet/examples.html>

Пример создания Excel документа:

```
HSSFWorkbook wb = new HSSFWorkbook();  
HSSFSheet sheet = wb.createSheet("new sheet");  
  
HSSFRow row = sheet.createRow((short)0);  
row.createCell((short)0).setCellValue("HelloWorld");  
  
FileOutputStream fileOut = new FileOutputStream("workbook.xls");  
wb.write(fileOut);  
fileOut.close();
```



# Взаимодействие с окружением

- <http://java.sun.com/javase/6/docs/api/java/awt/Desktop.html>
- [http://java.sun.com/developer/technicalArticles/J2SE/Desktop/javase6/desktop\\_api/](http://java.sun.com/developer/technicalArticles/J2SE/Desktop/javase6/desktop_api/)
- <https://jdic.dev.java.net/>

## Открыть файл или веб-страницу в браузере:

```
private void onLaunchBrowser(java.awt.event.ActionEvent evt) {  
    URI uri = null;  
    try {  
        uri = new URI(txtBrowserURI.getText());  
        desktop.browse(uri);  
    }  
    catch(IOException ioe) {  
        ioe.printStackTrace();  
    }  
    catch(URISyntaxException use) {  
        use.printStackTrace();  
    }  
    ...  
}
```

# Открыть e-mail клиент с указанием адреса назначения:


```
private void onLaunchMail(java.awt.event.ActionEvent evt) {  
    String mailTo = txtMailTo.getText();  
    URI uriMailTo = null;  
    try {  
        if (mailTo.length() > 0) {  
            uriMailTo = new URI("mailto", mailTo, null);  
            desktop.mail(uriMailTo);  
        } else {  
            desktop.mail();  
        }  
    }  
    catch (IOException ioe) {  
        ioe.printStackTrace();  
    }  
    catch (URISyntaxException use) {  
        use.printStackTrace();  
    }  
    ...  
}
```

# Открыть, редактировать, напечатать документ ассоциированной программой:

```
private void onLaunchDefaultApplication(java.awt.event.ActionEvent evt) {  
    String fileName = txtFile.getText();  
    File file = new File(fileName);  
  
    try {  
        switch(action) {  
            case OPEN:  
                desktop.open(file);  
                break;  
            case EDIT:  
                desktop.edit(file);  
                break;  
            case PRINT:  
                desktop.print(file);  
                break;  
        }  
    }  
    catch (IOException ioe) {  
        ioe.printStackTrace();  
    }  
    ...  
}
```



# Запуск сторонних приложений

- <http://java.sun.com/javase/6/docs/api/java/lang/Runtime.html>
- <http://www.javaworld.com/javaworld/jw-12-2000/jw-1229-traps.html> 

Простое, понятное, **неправильное** решение:

```
Runtime rt = Runtime.getRuntime();  
Process proc = rt.exec("javac");  
int exitVal = proc.exitValue(); //may not work!!!
```



# Обработка ошибок Отладка и Профайлинг

## Перехват исключений возникающих в интерфейсе

SUN/IBM JRE специфичное решение:☰

```
public class AWTExceptionHandler
{
    public static void register()
    {
        System.setProperty("sun.awt.exception.handler",
            AWTExceptionHandler.class.getName());
    }

    public void handle(Throwable ex)
    {
        ex.printStackTrace(System.out);
    }
}
```

## Решение использующее стандартную функциональность потоков:

```
class SwingExceptionHandler implements Thread.UncaughtExceptionHandler
{
    private static SwingExceptionHandler INSTANCE = new
SwingExceptionHandler();

    public static void register()
    {
        SwingUtilities.invokeLater(new Runnable() {
            @Override
            public void run() {
                Thread.currentThread().setUncaughtExceptionHandler(INSTANCE);
            }
        });
    }

    @Override
    public void uncaughtException(Thread thread, Throwable ex) {
        ex.printStackTrace(System.out);
        //reregister (each thread can be used only once, so EDT is restarted
in new Thread, so we should stick again to new one)
        register();
    }
}
```



# Профайлинг приложения в NetBeans

NetBeans встроен отличный профайлер. Хорошая идея — периодически запускать приложение под профайлером что бы отслеживать узкие места в производительности приложения.

The screenshot shows the NetBeans IDE 6.7.1 interface with the Profiler window open. The Profiler window displays a list of hot spots (methods) and their performance metrics. The left sidebar shows the Profiler controls, including a status section (CPU, Analyze Performance, Running) and a profiling results section (Take Snapshot, Live Results, Reset Collected Results). The right sidebar shows a pie chart of method categories (IO: 0.0%, UI: 36.9%, Project: 63.1%) and a section for HTTP request tracking (No data available).

Hot Spots - Method	Self time	Self time	Invocations
edu.umd.cs.piccolo.nodes.PText.recomputeLayout ()	5506 ms (33,8%)	6	1
morenoapp.components.thinktable.ThinktableMainJPanel.initComponents	4664 ms (28,6%)	1	1
morenoapp.components.welcome.WelcomeMainJPanel.<init> ()	749 ms (4,6%)	1	1
morenoapp.components.welcome.WelcomeMainJPanel.initComponents..	662 ms (4,1%)	1	1
org.jdesktop.application.SingleFrameApplication.initRootPaneContainer	352 ms (2,2%)	1	1
org.jdesktop.application.LocalStorage.load (String)	241 ms (1,5%)	1	1
edu.umd.cs.piccolo.nodes.PPath.getPathBoundsWithStroke ()	196 ms (1,2%)	6	1
morenoapp.components.common.AbstractMorenoTableModel.fireTableCh	179 ms (1,1%)	107	1
morenoapp.components.input.InputRespondentsPanel.initComponents..	169 ms (1%)	1	1
org.jdesktop.application.TextActions.<init> (org.jdesktop.application.A...	130 ms (0,8%)	1	1
org.jdesktop.application.ApplicationActionMap.addAnnotationActions...	110 ms (0,7%)	6	1
morenoapp.components.thinkgraphv2.components.CommonDisplayConfigJP	92.1 ms (0,6%)	1	1
org.jdesktop.application.ApplicationContext.<init> ()	86.5 ms (0,5%)	1	1
morenoapp.components.common.HintTextField.paint (java.awt.Graphics)	80.1 ms (0,5%)	8	1
morenoapp.components.config2.Configuration2JPanel.initComponents..	78.2 ms (0,5%)	1	1
edu.umd.cs.piccolo.util.PUtil.createBasicSceneGraph ()	76.3 ms (0,5%)	1	1
morenoapp.components.thinktable.ThinktableQuestionsFilterJPanel.apply	62.1 ms (0,4%)	3	1
morenoapp.components.workplace.TabbedWorkplace.<init> (morenoa...	61.7 ms (0,4%)	1	1
morenoapp.components.config2.QuestionsListEditorJPanel.paint (java...	60.6 ms (0,4%)	1	1
com.jgoodies.binding.beans.BeanUtils.getPropertyDescriptor (Class,...	59.4 ms (0,4%)	29	1
morenoapp.components.common.HintTextField.<init> ()	59.4 ms (0,4%)	4	1
morenoapp.util.MorenoUtil.<clinit>	57.9 ms (0,4%)	1	1
morenoapp.components.workplace.Workplace.<init> ()	56.1 ms (0,3%)	1	1
org.jdesktop.el.BeanELResolver\$BeanProperties.<init> (Class)	55.9 ms (0,3%)	6	1
org.jdesktop.application.SingleFrameApplication.show (org.jdesktop.ap...	52.8 ms (0,3%)	1	1
morenoapp.components.workplace.WizardPanel\$4.onSelectedView (m...	49.5 ms (0,3%)	5	1
edu.umd.cs.piccolo.nodes.PText.computeNextLayout (java.awt.font...	49.0 ms (0,3%)	9	1
morenoapp.components.config2.Configuration2JPanel.initLayout ()	48.3 ms (0,3%)	1	1
morenoapp.util.I18N.getResource (String)	47.7 ms (0,3%)	5	1
org.jdesktop.beansbinding.ELProperty.getBeanInfo (Object)	47.7 ms (0,3%)	17	1
morenoapp.components.thinkgraphv2.Thinkgraphv2JPanel.initCustom ()	46.5 ms (0,3%)	1	1
org.mb.listeners.ReflistenerManager.<init> (Class, ClassLoader)	45.0 ms (0,3%)	3	1
morenoapp.components.workplace.WizardPanel.<init> (morenoapp.co...	44.9 ms (0,3%)	1	1
morenoapp.components.common.OfficeColorsComboBox.<clinit>	44.4 ms (0,3%)	1	1
morenoapp.components.input.InputMainJPanel.initComponents ()	44.3 ms (0,3%)	1	1
morenoapp.model.app.MorenoApplicationData.<init> ()	44.0 ms (0,3%)	1	1
morenoapp.model.app.MorenoApp.initialize (String)	43.6 ms (0,3%)	1	1

# JRebel — быстрое исправление ошибок

- <http://www.zereturnaround.com/jrebel/>
- <http://www.zereturnaround.com/jrebel/comparison/>

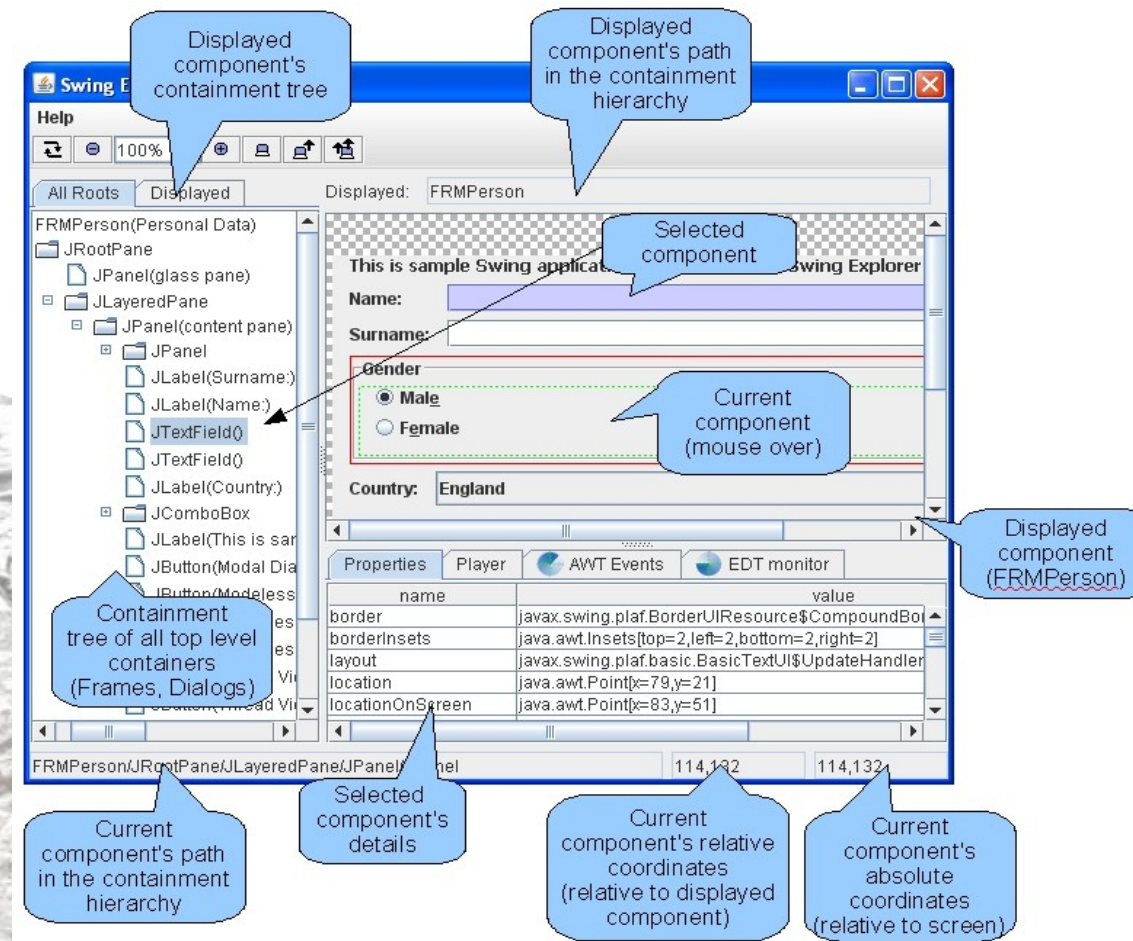
JRebel Feature Comparison Matrix

	Hot Redeploy	JVM HotSwap	JRebel
Time to reload	30s-15min	< 1s	< 1s
Leaks memory	Yes	No	No
Changes to method bodies	+	+	+
<b>IDE support</b>			
Eclipse	+	+	(plugin)
IntelliJ IDEA	+	+	(plugin)
NetBeans	+	+	(plugin)
<b>Changes to class structure</b>			
Adding/removing methods	+	-	+
Adding/removing constructors	+	-	+
Adding/removing fields	+	-	+
Adding/removing classes	+	-	+
Adding/removing annotations	+	-	+
Changing interfaces	+	-	+
Replacing superclass	+	-	-
Adding/removing implemented interfaces	+	-	-



# SwingExplorer – структура форм и отладка paint()

- <https://swingexplorer.dev.java.net/>

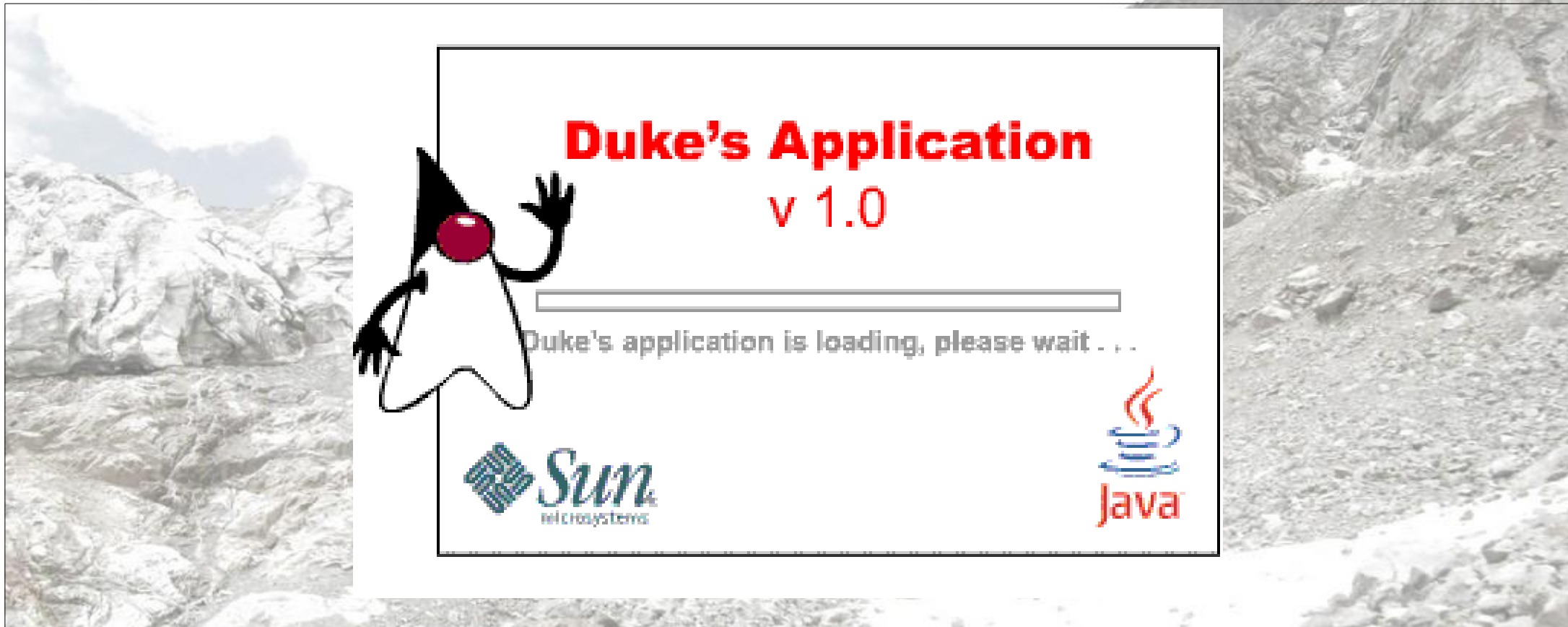




# Запуск приложения

## Splash Screen

- <http://java.sun.com/docs/books/tutorial/uiswing/misc/splashscreen.html>



# Установка splashscreen для приложения:

Через параметры виртуальной машины:

```
java -splash:<file name> <class name>
```

Через метаданные jar файла:

Manifest-Version: 1.0

Main-Class: <class name>

SplashScreen-Image: <image name>



# Пример отображения прогресса загрузки приложения

```
private static void startStartupProgress() {  
    new SwingWorker() {  
        @Override  
        protected Object doInBackground() throws Exception {  
            final SplashScreen ss = SplashScreen.getSplashScreen();  
            if (ss != null) {  
                Graphics g = ss.createGraphics();  
                int progress = 0;  
                while (ss.isVisible() == true) {  
                    g.setColor(Color.ORANGE);  
                    progress += 10;  
                    g.fillRect(0, 270, progress, 3);  
                    ss.update();  
                    Thread.sleep(250);  
                }  
                this.done();  
            }  
            return "OK";  
        }  
    }.execute();  
}
```



# Запуск Java приложения средствами JRE

<http://www.javalobby.org/articles/java2exe/>

## Запуск из командной строки:

```
java -Xmx200m -cp whatever.jar -Dsome.property MyApp
```

## Исполняемый Jar-файл:

```
Main-Class: MyAppMain  
Class-Path: mylib.jar
```

**Если приложение состоит из нескольких jar файлов, их можно собрать в один используя:**

- <http://one-jar.sourceforge.net/>
- <http://code.google.com/p/jarjar/>

# Создание EXE файла для запуска приложения

- <http://launch4j.sourceforge.net/>
- <http://www.ej-technologies.com/products/exe4j/overview.html>
- <http://izpack.org/>

## Конфигурационный файл для launch4j:

```
<launch4jConfig>
  <dontWrapJar>true</dontWrapJar>
  <headerType>gui</headerType>
  <jar></jar>
  <outfile>MyApplication.exe</outfile>
  <errTitle></errTitle>
  <cmdLine></cmdLine>
  <chdir></chdir>
  <priority>normal</priority>
  <downloadUrl>http://java.com/download</downloadUrl>
  <supportUrl>http://my-website.com</supportUrl>
  <customProcName>true</customProcName>
  <stayAlive>false</stayAlive>
  <manifest></manifest>
  <icon>logo-16.ico</icon>
  <classPath>
    <mainClass>my.app.Main</mainClass>
    <cp>myapp.jar</cp>
  </classPath>
  <jre>
    <path>jre</path>
    <minVersion>1.6.0</minVersion>
    <maxVersion></maxVersion>
    <jdkPreference>jreOnly</jdkPreference>
    <opt>-splash:resources/splash.png</opt>
  </jre>
</launch4jConfig>
```

## Ключевые моменты для launch4j:

1. Можно создать отдельный EXE файл, который будет запускать jar файлы, либо
2. Обернуть jar файл в EXE так что приложение будет выглядеть одним файлом
3. В каталог с приложением можно положить JRE и задать в опциях использовать именно эту JRE. Так что пользователю будет не обязательно иметь установленную Java нужной версии.

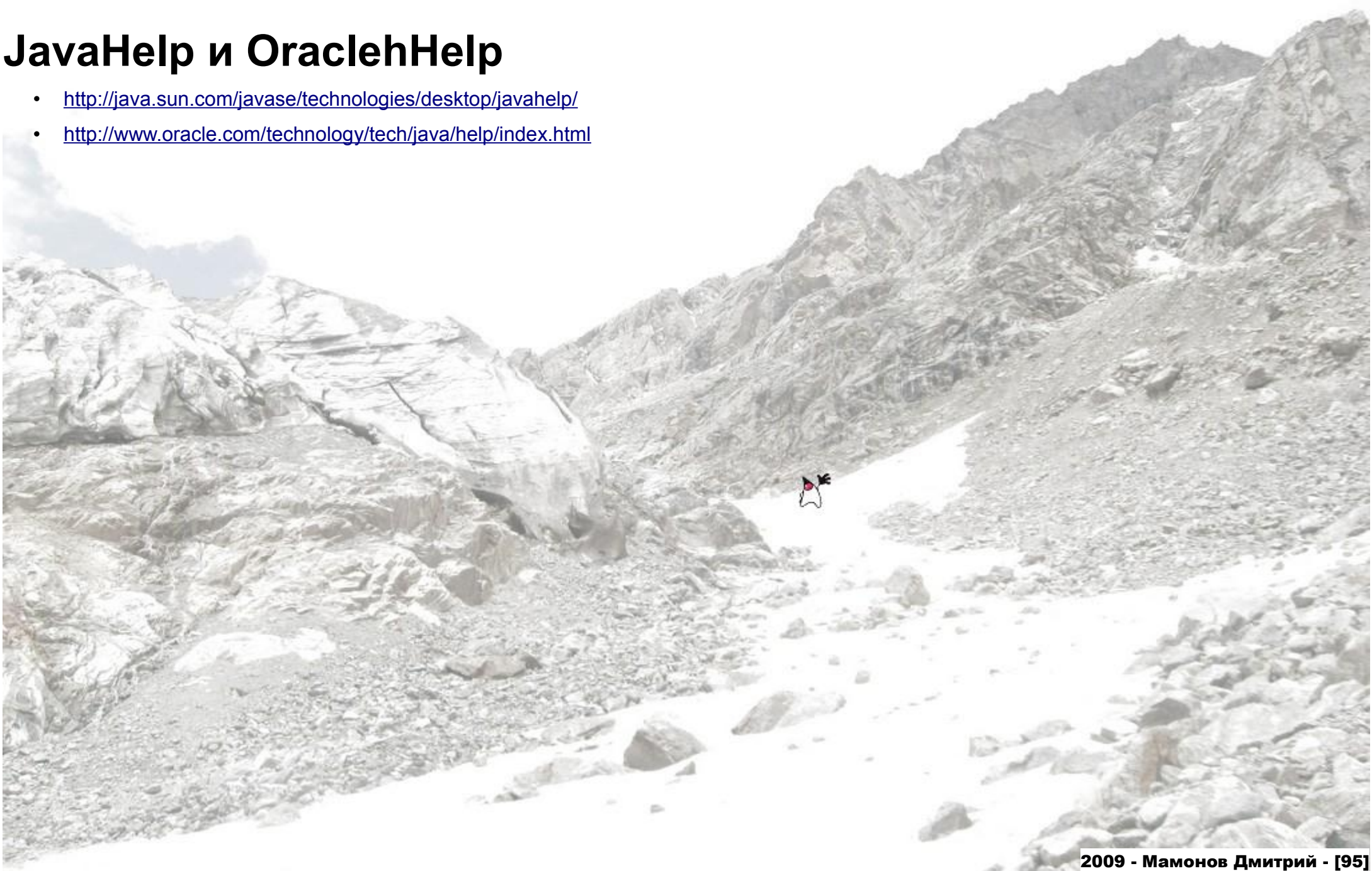
IzPack так же содержит ряд утилит для обёртки Java приложений в исполняемые файлы, в том числе для Mac OS.



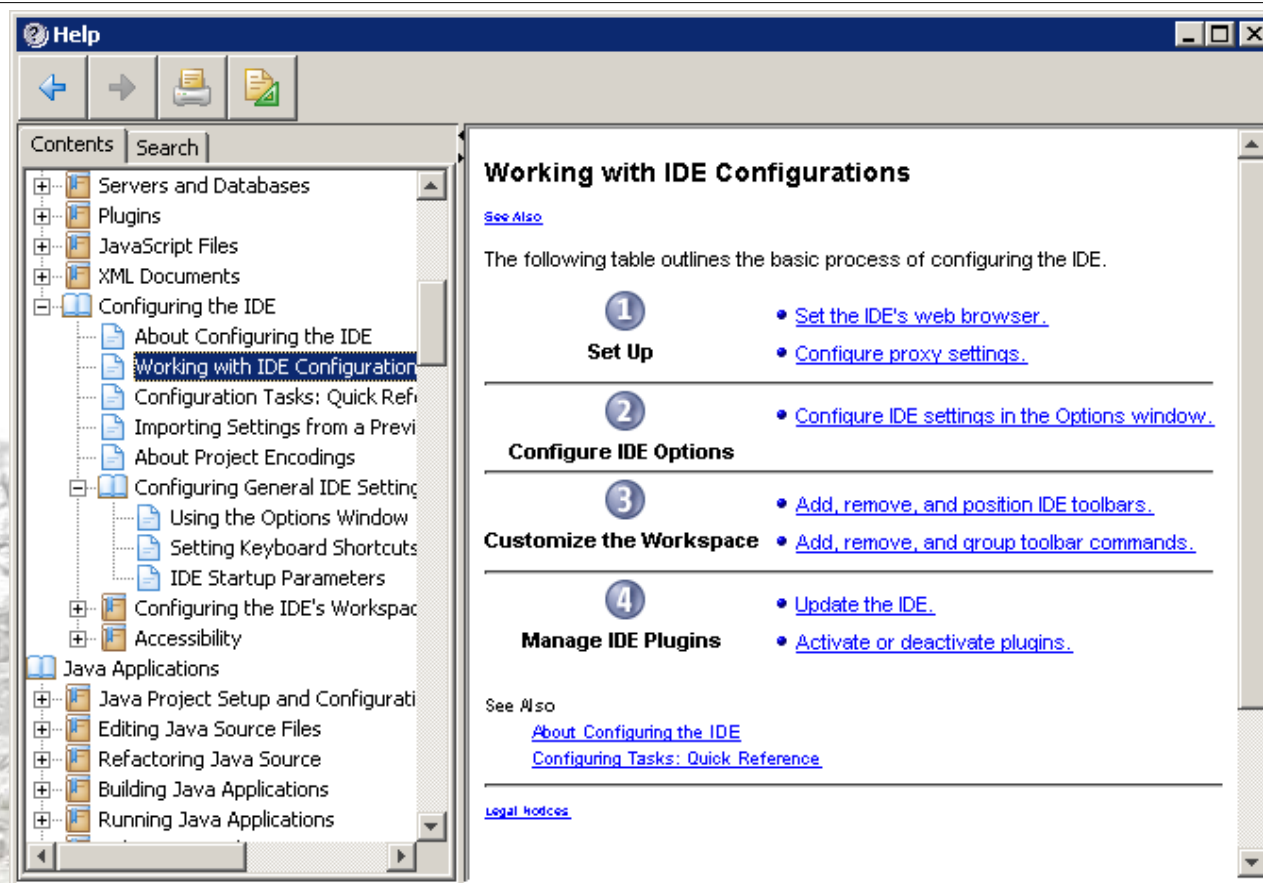
# Документация и справка

## JavaHelp и OraclehHelp

- <http://java.sun.com/javase/technologies/desktop/javahelp/>
- <http://www.oracle.com/technology/tech/java/help/index.html>

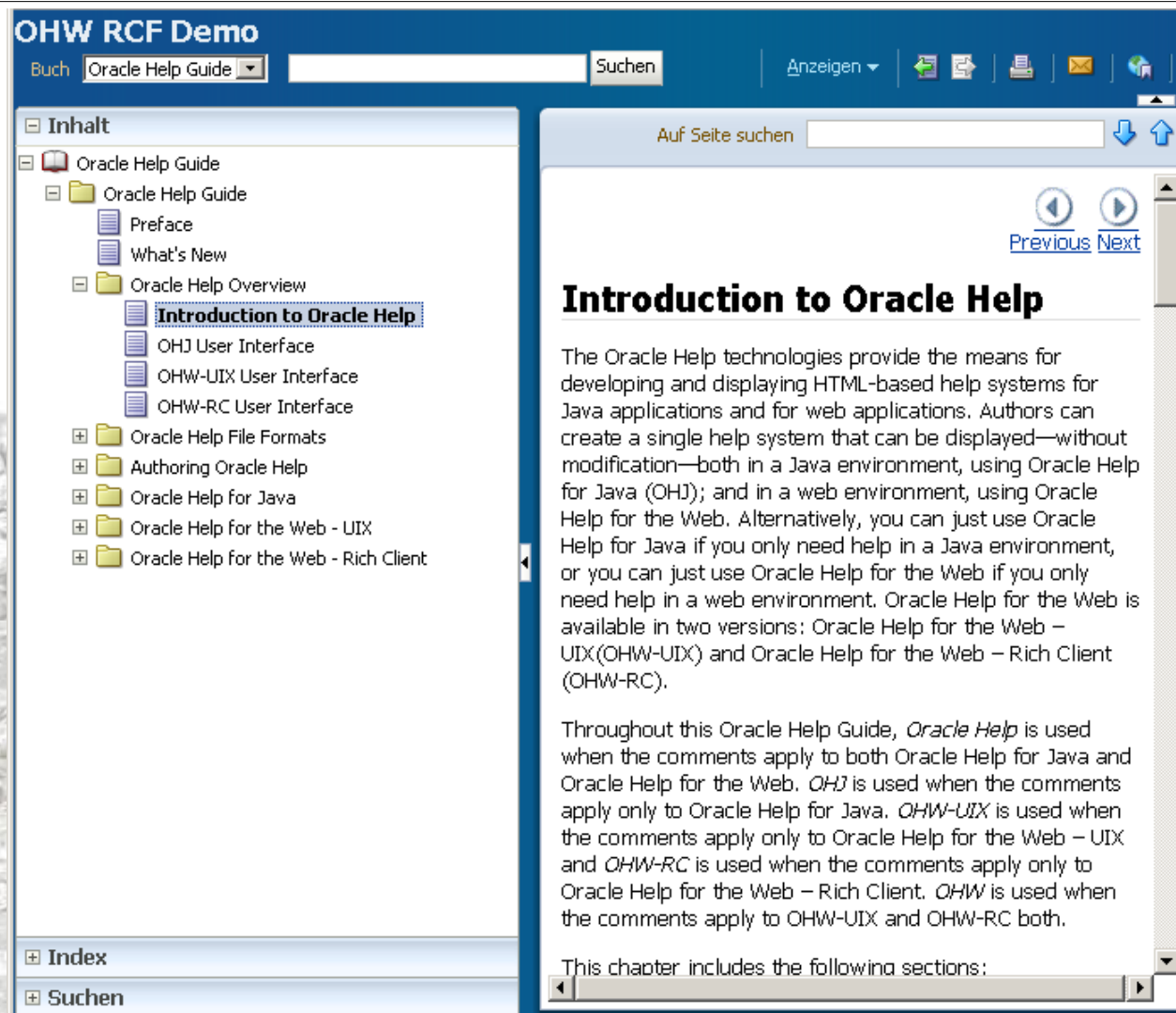


# Вид окна справки JavaHelp:☰



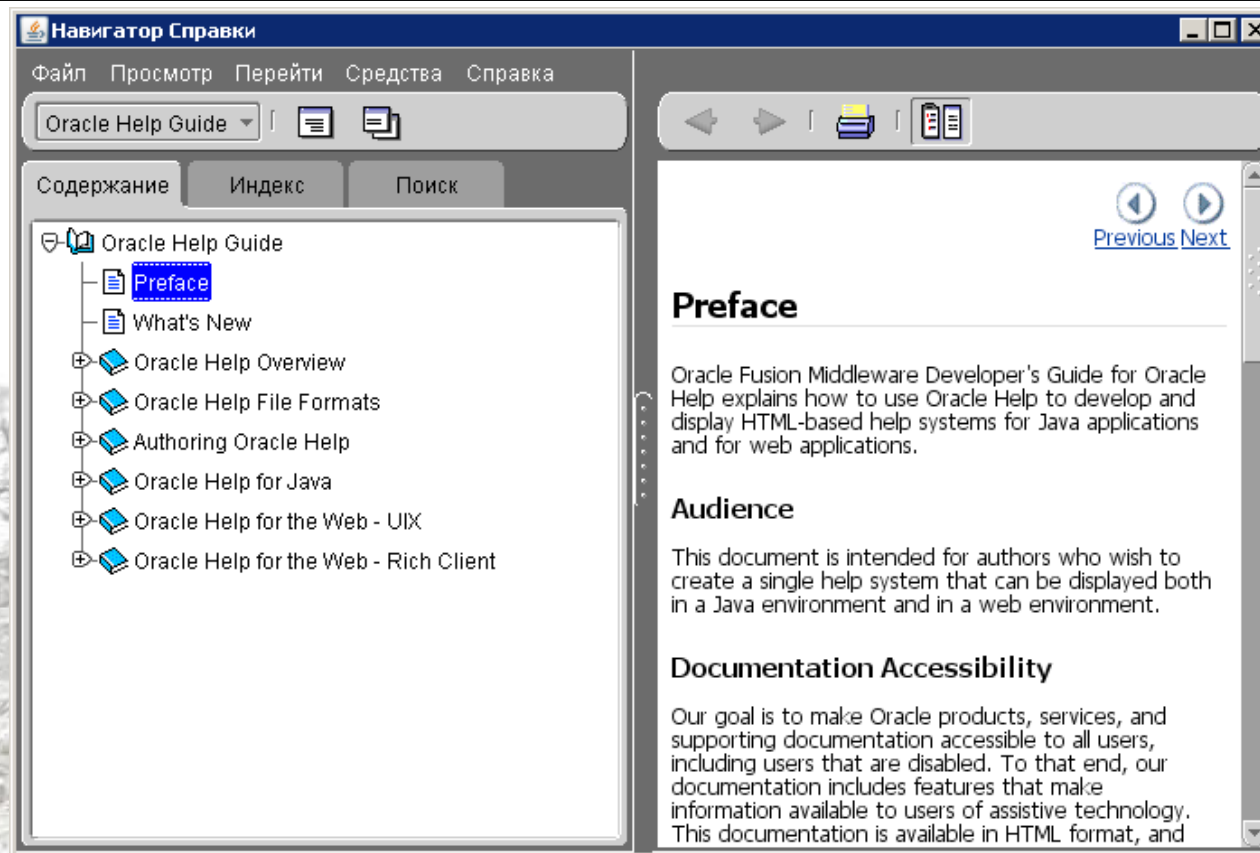


# OracleHelp (web-версия):





# OracleHelp (Desktop версия):



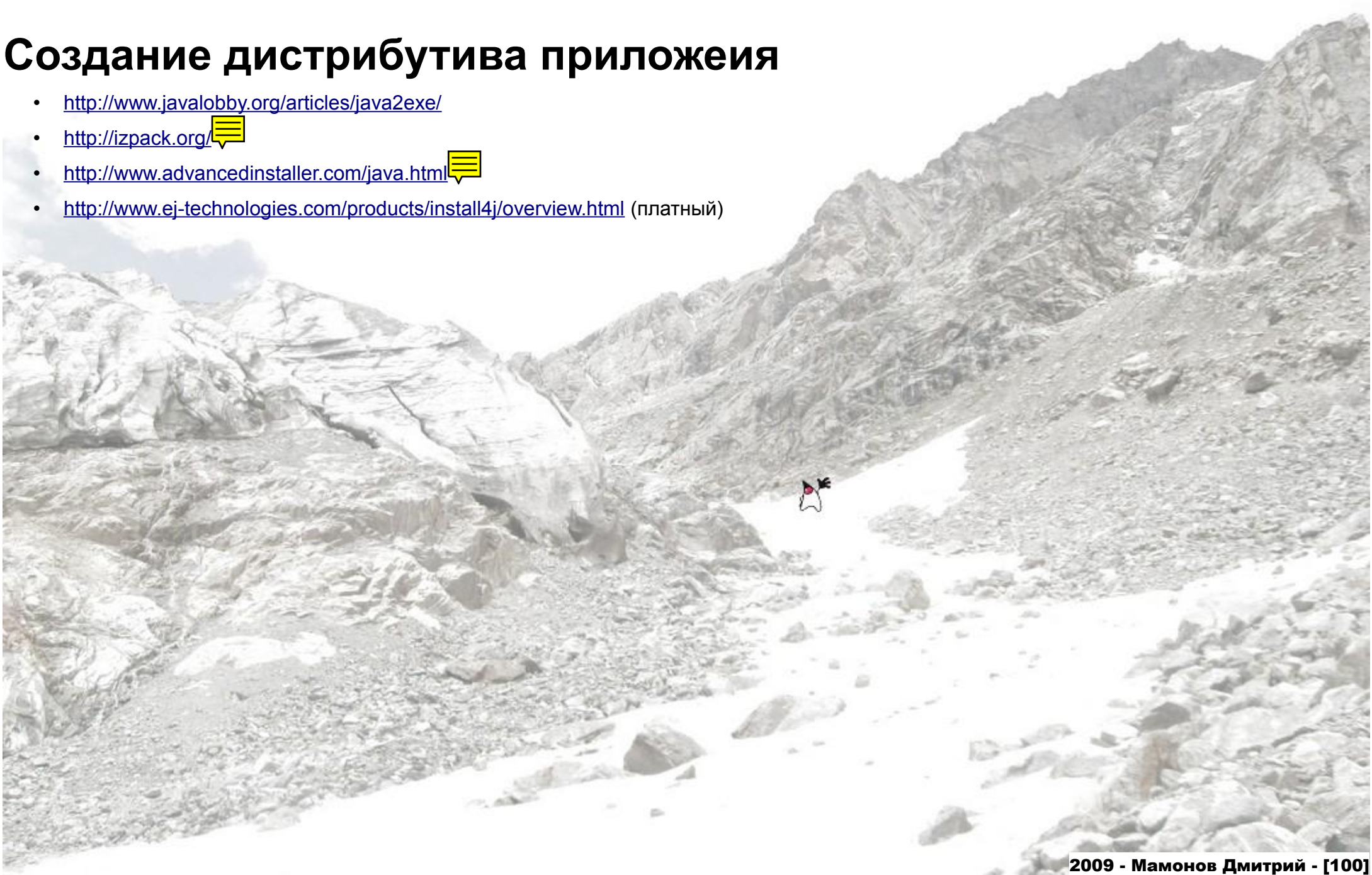




# Распространение

## Создание дистрибутива приложения

- <http://www.javalobby.org/articles/java2exe/>
- <http://izpack.org/> 
- <http://www.advancedinstaller.com/java.html> 
- <http://www.ej-technologies.com/products/install4j/overview.html> (платный)





# Создание WebStart приложения

- <http://java.sun.com/docs/books/tutorial/deployment/webstart/index.html>

## WebStart дескриптор приложения:

```
<?xml version="1.0" encoding="UTF-8"?>
<jnlp spec="1.0+" codebase="" href="">
  <information>
    <title>Dynamic Tree Demo</title>
    <vendor>Dynamic Team</vendor>
  </information>
  <resources>
    <!-- Application Resources -->
    <j2se version="1.6+"
      href="http://java.sun.com/products/autodl/j2se"/>
    <jar href="DynamicTreeDemo.jar" main="true" />

  </resources>
  <application-desc
    name="Dynamic Tree Demo Application"
    main-class="webstartComponentArch.DynamicTreeApplication"
    width="300"
    height="300">
  </application-desc>
  <update check="background"/>
</jnlp>
```

# Создание Applet

- <http://java.sun.com/docs/books/tutorial/deployment/applet/index.html>
- <http://java.sun.com/developer/technicalArticles/javase/java6u10/#dt>
- [http://blogs.sun.com/thejavatutorials/entry/deployment\\_toolkit\\_101](http://blogs.sun.com/thejavatutorials/entry/deployment_toolkit_101)

## Запуск Applet/WebStart приложения при помощи JavaScript:

```
<script src="http://java.com/js/deployJava.js"></script>

<script>
  deployJava.runApplet({codebase:"http://www.example.com/applets/",
    archive:"ExampleApplet.jar", code:"Main.class",
    width:"320", Height:"400"}, null, "1.6");
</script>
```

## Достоинства:

1. Автоматическое определение установленной версии Java
2. Прозрачное для пользователя обновление Java
3. Простой способ запустить Applet/WebStart приложение одной строчкой JavaScript.



# Другие полезные статьи

- [http://www.jroller.com/DhilshukReddy/entry/swing\\_resources](http://www.jroller.com/DhilshukReddy/entry/swing_resources)
- <http://karussell.wordpress.com/2009/10/08/java-application-frameworks-not-only-client-side/>
- <http://lopeathal.wikispaces.com/Open+Source+Desktop+Libraries>

