

08. Создание GUI

Лекции по информатике
для студентов второго курса Высшей школы ИТИС КФУ
2020

Ференец Александр Андреевич

старший преподаватель кафедры программной инженерии

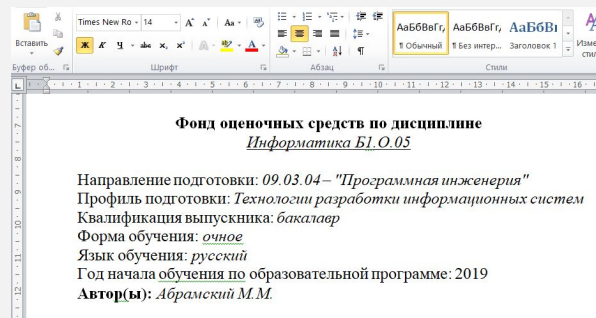
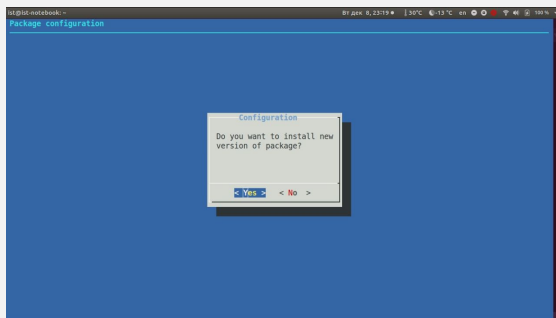
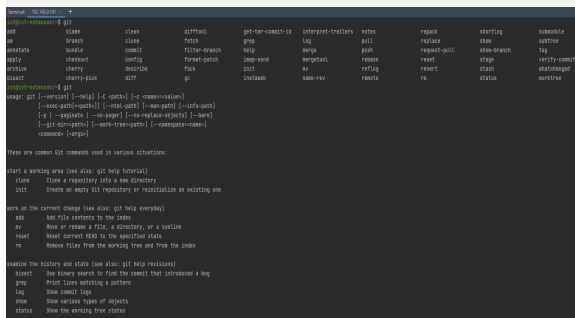
С использованием материалов
к. т. н., доцента кафедры программной инженерии Абрамского М.М.

aferenets@it.kfu.ru

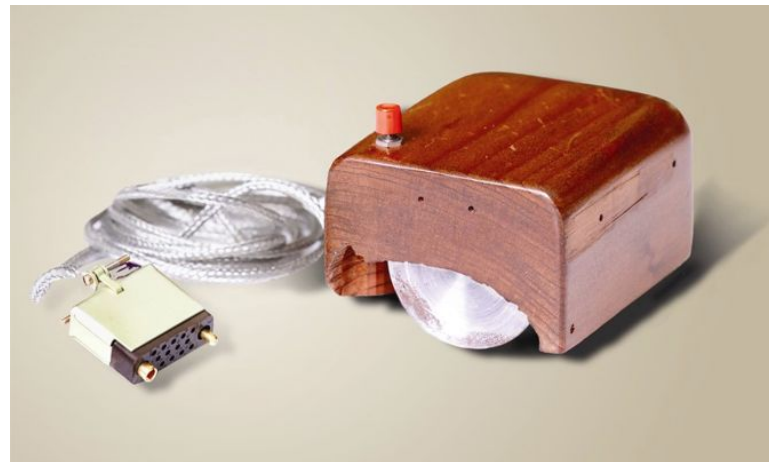
Graphical User Interface

Интерфейс – совокупность аппаратно-программных средств, обеспечивающая обмен данных между исполнительными устройствами автоматической системы или между человеком и машиной.

GUI – интерфейс машины для человека, основанный на графических элементах.



ИСТОРИЯ GUI. Взаимодействие с ПК



Компьютерный терминал — устройство, используемое для взаимодействия пользователя (или оператора) с компьютером или компьютерной системой, локальной или удалённой. *Не обязательно текст!*

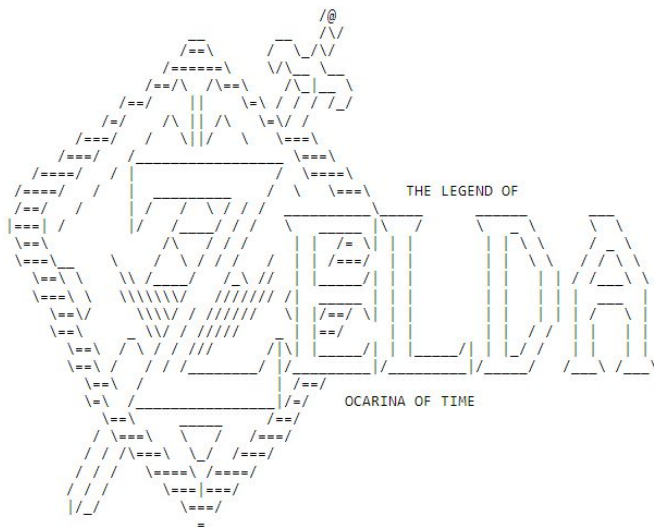
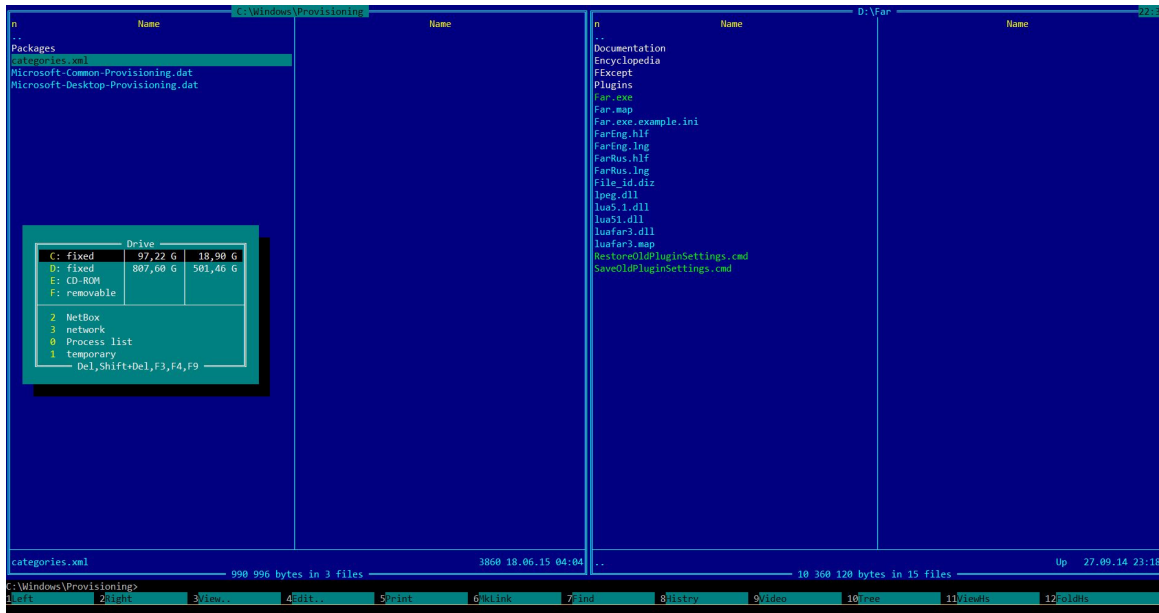
Можно подразумевать ПО для непосредственного ввода команд. В современных *nix-системах используются программы Виртуальные терминалы, которые запускаются в графической среде и выполняют те же операции, что более “низкоуровневые”.








Не бойтесь текстового терминала! Он может быстрее мыши в ряде ситуаций!

Существуют различные оболочки (например, bash), которые:

- Запоминают историю команд;
- Предлагают автодополнение;
- Поддерживают написание мини-скриптов.
- А ещё Unix-way!

ИСТОРИЯ GUI. Псевдографический интерфейс

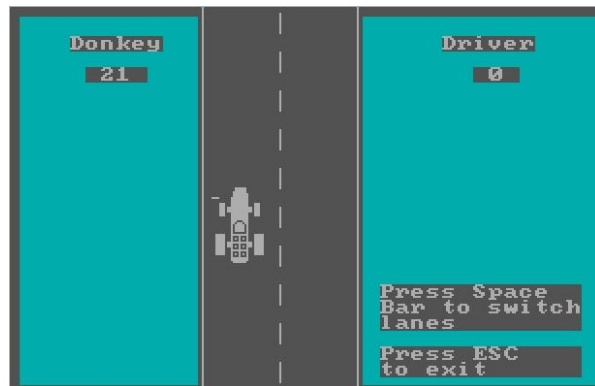


- ASCII code 191 =  (Box drawing character single line upper right corner)
- ASCII code 192 =  (Box drawing character single line lower left corner)
- ASCII code 193 =  (Box drawing character single line horizontal and up)
- ASCII code 194 =  (Box drawing character single line horizontal down)
- ASCII code 195 =  (Box drawing character single line vertical and right)
- ASCII code 196 =  (Box drawing character single horizontal line)
- ASCII code 197 =  (Box drawing character single line horizontal vertical)

Bill Gates, Neil Konzen, 1981

```
1080 COLOR 15,0:LOCATE 17,4,0:PRINT "(C) Copyright IBM Corp 1981, 1982"  
1090 COLOR 14,0:LOCATE 23,7,0:PRINT "Press space bar to continue"  
1100 IF INKEY$<>" " THEN GOTO 1100  
1110 CMD$ = INKEY$  
1120 IF CMD$ = " " THEN GOTO 1110  
1130 IF CMD$ = CHR$(27) THEN GOTO 1298  
1140 IF CMD$ = " " THEN GOTO 1160  
1150 GOTO 1110  
1160 DEF SEG=0  
1170 IF (PEEK(&H410) AND &H30)<>&H30 THEN DEF SEG:GOTO 1291  
1180 WIDTH 80:CLS:LOCATE 3,1
```

<https://github.com/coding-horror/donkey.bas/blob/master/donkey.bas>



1985, Windows 1.0

Первоначально – лишь графическая надстройка над MS-DOS

А ещё раньше...

1985, Windows 1.0

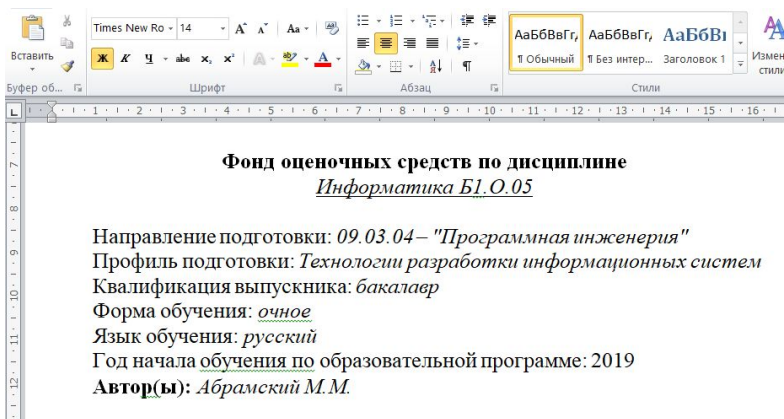
Первоначально – лишь графическая надстройка над MS-DOS

Xerox PC 1973

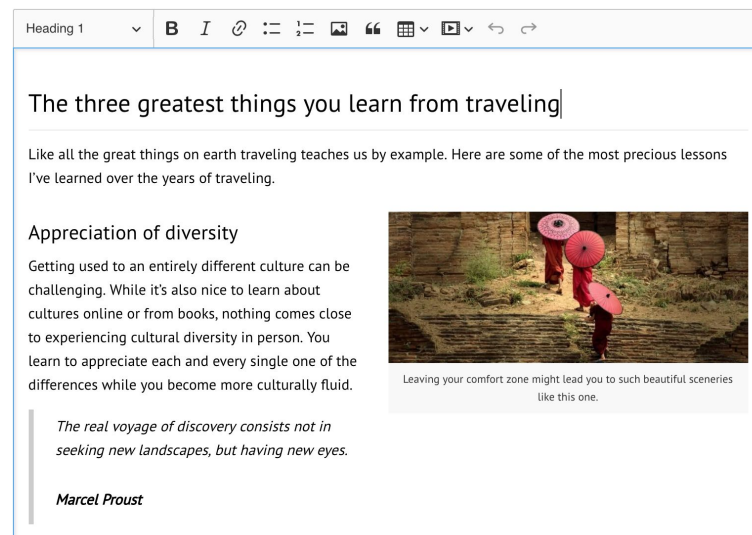
Вообще многим вещам мы обязаны компаниям Xerox, HP.

Но они их не смогли успешно внедрить, а потом MS, Apple представили миру “инновации”.





MS Word



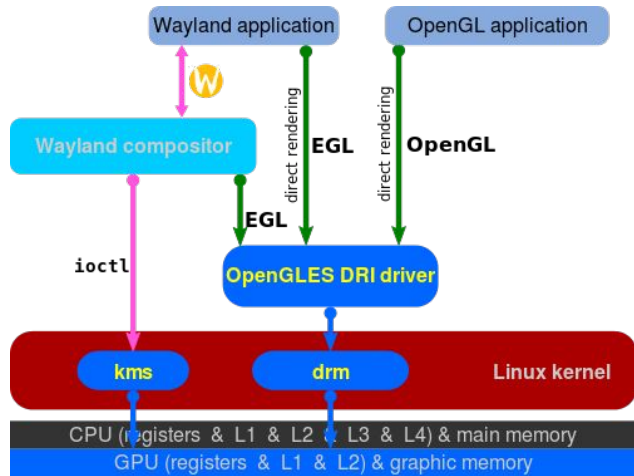
CKEditor

GUI. Из чего состоит?



1. Компоненты
 - a. Управляющие элементы
 - b. Каркас (Layout)
 - c. Медиа
 - d. Прочее
2. Подписка на события
3. Рисование и анимация
4. Управление потоками
5. Интеграция (в браузер, ОС и в другие системы создания GUI)
6. * Описание стилей?

- Desktop Window Manager, GDI, ...
- X Window System (X11)
 - + Compiz +...
- ~Wayland protocol



- QT и QT Lambi в Java
- Swing
- JavaFX
- Windows Forms и Windows Presentation Foundation
- XUL
- Delphi и Lazarus (последнее обновление на 12.2020 – 06.2020)

MVC? MVVM?

Главное: Разделение логики и содержимого!

Легковесная структура <-> Комбайн

Библиотека <-> Фреймворк

Часто вводят язык разметки для описания вида:

- Markdown

- HTML

- FXML

- XUL

- JSX?

```
const element = (  
  <h1 className="greeting">  
    Привет, {title}!  
  </h1>  
);
```

Headers

```
# This is an <h1> tag  
## This is an <h2> tag  
##### This is an <h6> tag
```

Emphasis

```
*This text will be italic*  
_This will also be italic_  
  
**This text will be bold**  
__This will also be bold__  
  
_You **can** combine them_
```

```
<?xml version="1.0"?>  
<?xml-stylesheet href="chrome://global/skin/" type="text/css"?>  
  
<window id="vbox example" title="Example 3..."  
  xmlns="http://www.mozilla.org/keymaster/gatekeeper/there.is.only.xul">  
  <vbox>  
    <button id="yes" label="Yes"/>  
    <button id="no" label="No"/>  
    <button id="maybe" label="Maybe"/>  
  </vbox>  
</window>
```

- Controller в JavaFX
- ZK Composer
- Listener в AWT

```
package org.zkoss.simple;

// some import statements are omitted for brevity.

public class RegistrationComposer extends SelectorComposer<Component> {

    @Wire
    private Button submitButton;

    @Wire
    private Checkbox acceptTermBox;

}
```

```
public class SignInButtonListener implements ActionListener{
    @Override
    public void actionPerformed(ActionEvent e) {
        Button b = (Button) e.getSource();
        b.getParent().setBackground(Color.red);
    }
}
```

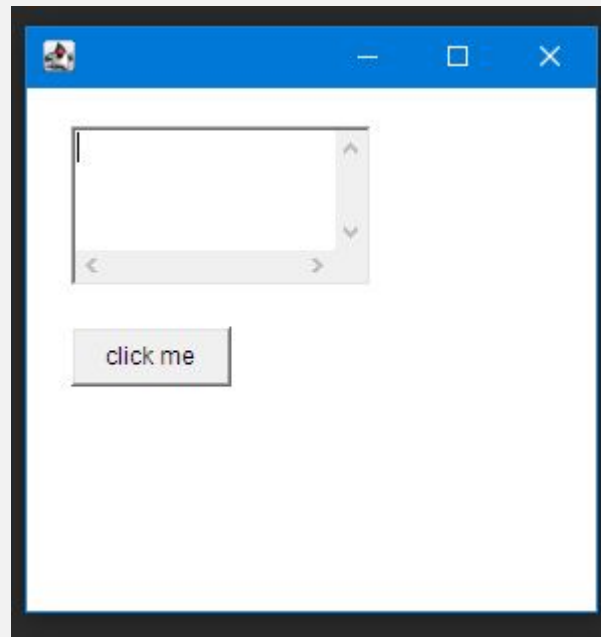
- Различные встраиваемые выражения в разметку

```
<button text="{parent()>children[last()]>getText()>eq('red')?'blue':'red'}">
```


Abstract Window Toolkit

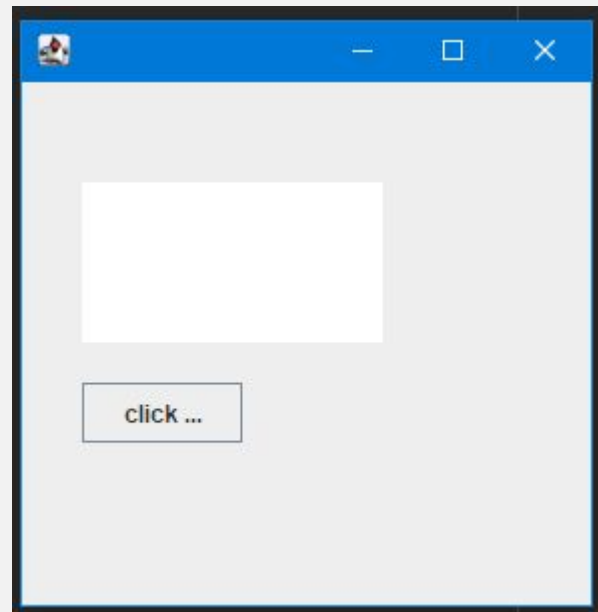
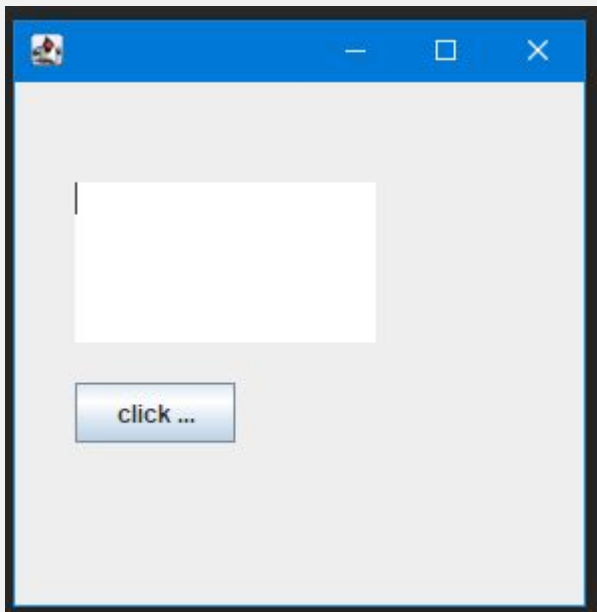
Абстракция и кроссплатформенность!

- Компоненты
- Подписка на события
- Layout Manager'ы



Фактически декоратор AWT:

Строится на основе, добавляя идею Native Look And Feel, улучшенную работу с потоками.



JavaFX Public API's and Scene Graph

Quantum Toolkit

Prism

Glass Windowing Toolkit

Media Engine

Web Engine

Java 2D

Open GL

D3D

Java Virtual Machine

Glass Windowing Toolkit: оконный интерфейс, кнопки, поля и т.п.

+ **Pulse:** диспетчер событий

Prism: работа с графикой

Quantum Toolkit: Абстракция над Prism и Glass вместе

Существует множество систем, которые манипулируют понятиями элементов форм, вывода данных, подпиской на события элементов управления.

*Например, **Google Web Toolkit**.*

Он позволяет генерировать клиентский код. Тем самым разработчик освобождается от необходимости два раза описывать объектную модель и ряд похожих процессов на клиенте и сервере.



Состоит из множества компонент:

QtCore, QtGui, QtWidgets, QtOpenGL, QtWebEngine, QtMultimedia и других

Обёртка в Java: *Qt Jambi*

+

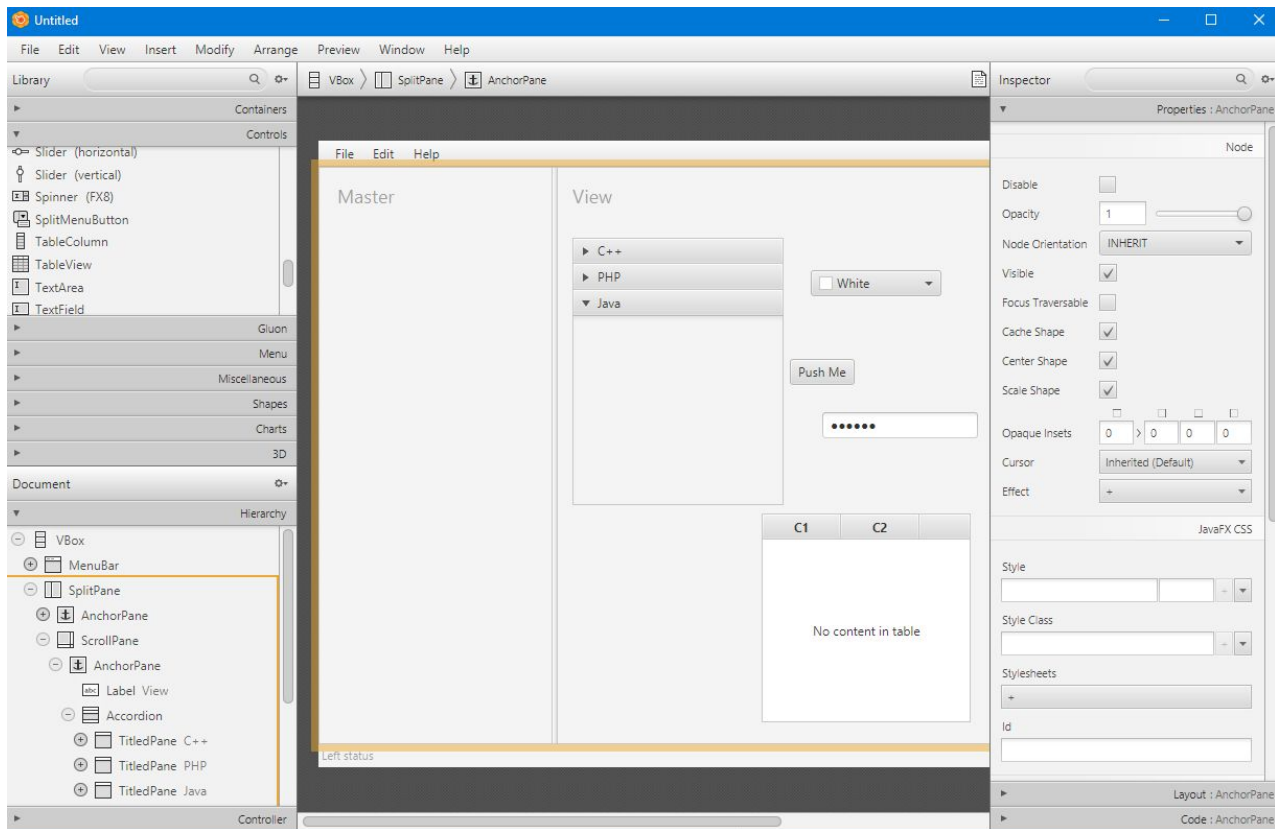
PyQt • PySide • QtRuby • **Qt Jambi** • PythonQt

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
  <class>Notepad</class>
  <widget class="QMainWindow" name="Notepad">
    <property name="geometry">
      <rect>
        <x>0</x>
        <y>0</y>
        <width>800</width>
        <height>400</height>
      </rect>
    </property>
    <property name="windowTitle">
      <string>Notepad</string>
    </property>
    <widget class="QWidget" name="centralWidget">
      <layout class="QVBoxLayout" name="verticalLayout">
        <item>
          <widget class="QTextEdit" name="textEdit"/>
        </item>
      </layout>
    </widget>
    <widget class="QMenuBar" name="menuBar">
```

Также предпринимались попытки выработать стандарты описания интерфейсов.

XUL – язык разметки для создания динамических пользовательских интерфейсов на основе XML. XUL разрабатывается в рамках проекта Mozilla и является частью платформы XULRunner.

- JavaFx Scene Builder
- QT Designer
- Delphi



Swing и **JavaFX** достаточно мощные технологии создания интерфейсов в Java.

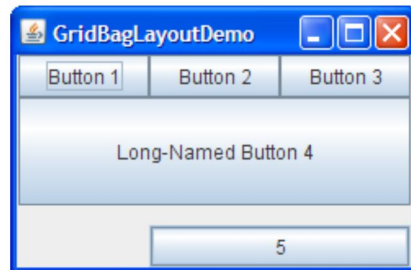
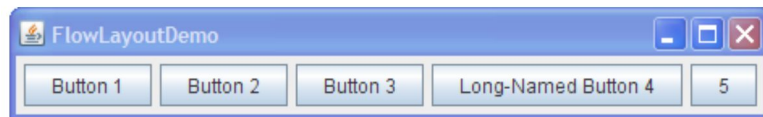
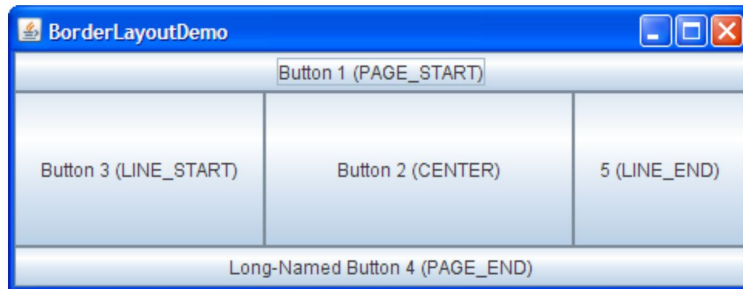
Swing вполне жизнеспособен и до сих пор используется некоторыми компаниями, даже развивается, но

JavaFx имеет более продуманную архитектуру, в некотором смысле быстрее и содержит больше инструментов.


```
JFrame frame = new JFrame( title: "Hello world title");  
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
  
JButton signInButton = new JButton( text: "Sign In");  
frame.getContentPane().add(signInButton);  
  
frame.setBounds( x: 300, y: 200, width: 1100, height: 800);  
frame.setVisible(true);
```

<https://docs.oracle.com/javase/tutorial/uiswing/layout/visual.html>

- BorderLayout
- BoxLayout
- CardLayout
- FlowLayout
- GridBagLayout
- GridLayout
- GroupLayout
- SpringLayout



<CODE>

Пример кода с подпиской на события через отдельный класс, реализующий `EventListener` определённого рода, через анонимный класс, лямбда-выражение. Важность правильного выбора типа события.

```
doNetworkRequestButton.addActionListener(e -> {  
    SwingWorker worker = new SwingWorker<String, Integer>() {  
        protected String doInBackground() throws Exception {  
            HttpResponse response = doRequest();// Do long network request here  
            return response.getBody();  
        }  
  
        protected void done() {  
            try {  
                respLabel.setText("Data has been updated:" + get());  
            }  
            catch (InterruptedException | ExecutionException ex){  
                respLabel.setText("Error has been occured:" + ex.getMessage());  
            }  
        }  
    };  
    worker.execute();  
});
```

<CODE>

Пример приложения с созданием окна, элементами в нём.

JavaFX может не входить в стандартную поставку JDK!

<https://www.jetbrains.com/help/idea/javafx.html#vm-options>

<https://www.jetbrains.com/help/idea/opening-fxml-files-in-javafx-scene-builder.html>

Для сборки проекта в Maven, Gradle есть специальные плагины.

```
<build>
  <plugins>
    <plugin>
      <groupId>org.openjfx</groupId>
      <artifactId>javafx-maven-plugin</artifactId>
      <version>0.0.3</version>
      <configuration>
        <mainClass>com.example.App</mainClass>
      </configuration>
    </plugin>
  </plugins>
</build>
```

```
public class Main extends Application {  
  
    @Override  
    public void start(Stage primaryStage) throws Exception{  
        Parent root = FXMLLoader.load(getClass().getResource( name: "sample.fxml"));  
        Scene s = new Scene(root, v: 300, v1: 275);  
        s.getStylesheets().add(getClass().getResource( name: "main.css").toExternalForm());  
        primaryStage.setTitle("Hello World");  
        primaryStage.setScene(s);  
        primaryStage.show();  
    }  
  
    public static void main(String[] args) { launch(args); }  
}
```

Stage <- Scene <- Root Node

JavaFX. Добавление компонент и Layout Manager'ы

```
VBox pane = new VBox();  
Label infoLabel = new Label();  
pane.getChildren().add(infoLabel);
```

Не перепутайте импорты с AWT!

```
import javafx.application.Application;  
import javafx.fxml.FXMLLoader;  
import javafx.scene.Parent;  
import javafx.scene.Scene;  
import javafx.scene.control.Button;  
import javafx.scene.control.Label;  
import javafx.scene.input.MouseEvent;  
import javafx.scene.layout.BorderPane;  
import javafx.scene.layout.VBox;  
import javafx.stage.Stage;
```



```
<?import javafx.scene.image.ImageView?>
<?import javafx.scene.image.Image?>
<?import javafx.scene.control.ScrollPane?>
<BorderPane fx:controller="sample.Controller" xmlns="http://javafx.com/javafx/11.0.1" xmlns:fx="http://javafx.com/fxml/1">
  <top>
    <BorderPane styleClass="header-section">
      <left>
        <Label id="header-text" text="Application Header"></Label>
      </left>
      <right>
        <Button id="account" text="Account" onAction="#accountClick">
          <graphic>
            <ImageView fitHeight="24" fitWidth="24" pickOnBounds="true" preserveRatio="true">
              <image>
                <Image url="@javafx_account_example.png" />
              </image>
            </ImageView>
          </graphic>
        </Button>
      </right>
    </BorderPane>
  </top>
  <left>
    <ScrollPane hbarPolicy="NEVER" vbarPolicy="AS_NEEDED" prefHeight="700">
      <content>
        <VBox styleClass="sidebar-section" fx:id="labelContainer">
          <children>
            <Label text="Sidebar Item1"></Label>
            <Label text="Sidebar Item2" fx:id="label2"></Label>
            <Label text="Sidebar Item3"></Label>
            <Label text="Sidebar Item4"></Label>
          </children>
        </VBox>
      </content>
    </ScrollPane>
  </left>
```

- Создать Контроллер и прикрепить его к сцене через `fx:controller`, прикрепить метод в нём в `fxml` через `on***`-атрибут
- В Java-коде вызвать метод `addEventHandler`

```
Service<Void> service = new Service<Void>() {  
    @Override  
    protected Task<Void> createTask() {  
        return new Task<Void>() {  
            @Override  
            protected Void call() throws Exception {  
                //Background work  
                final CountDownLatch latch = new CountDownLatch(1);  
                Platform.runLater(new Runnable() {  
                    @Override  
                    public void run() {  
                        try{  
                            //FX Stuff done here  
                        }finally{  
                            latch.countDown();  
                        }  
                    }  
                });  
                latch.await();  
                //Keep with the background work  
                return null;  
            }  
        };  
    }  
};  
service.start();
```

```
@Override
public void start(Stage primaryStage) {
    WebView webView = new WebView();
    webView.getEngine().load( "http://javafx.com" );
    root.getChildren().add( webView );
    primaryStage.setScene(new Scene(root, 600, 500));
    primaryStage.show();
}
```

<CODE>

Пример приложения с формой, где добавляются новые элементы и на них вешаются обработчики.