

Программирование на Java

7. Введение в GUI

Глухих Михаил Игоревич
mailto: glukhikh@mail.ru

GUI-концепции

- ▶ Программа **отвечает на запросы**, а не действует активно

GUI-концепции

- ▶ Программа **отвечает на запросы**, а не действует активно
 - Консольное приложение получает задачу и решает её

GUI–концепции

- ▶ Программа **отвечает на запросы**, а не действует активно
 - Консольное приложение получает задачу и решает её
 - GUI–приложение функционирует в режиме «вопрос (пользователя) – ответ (программы)»

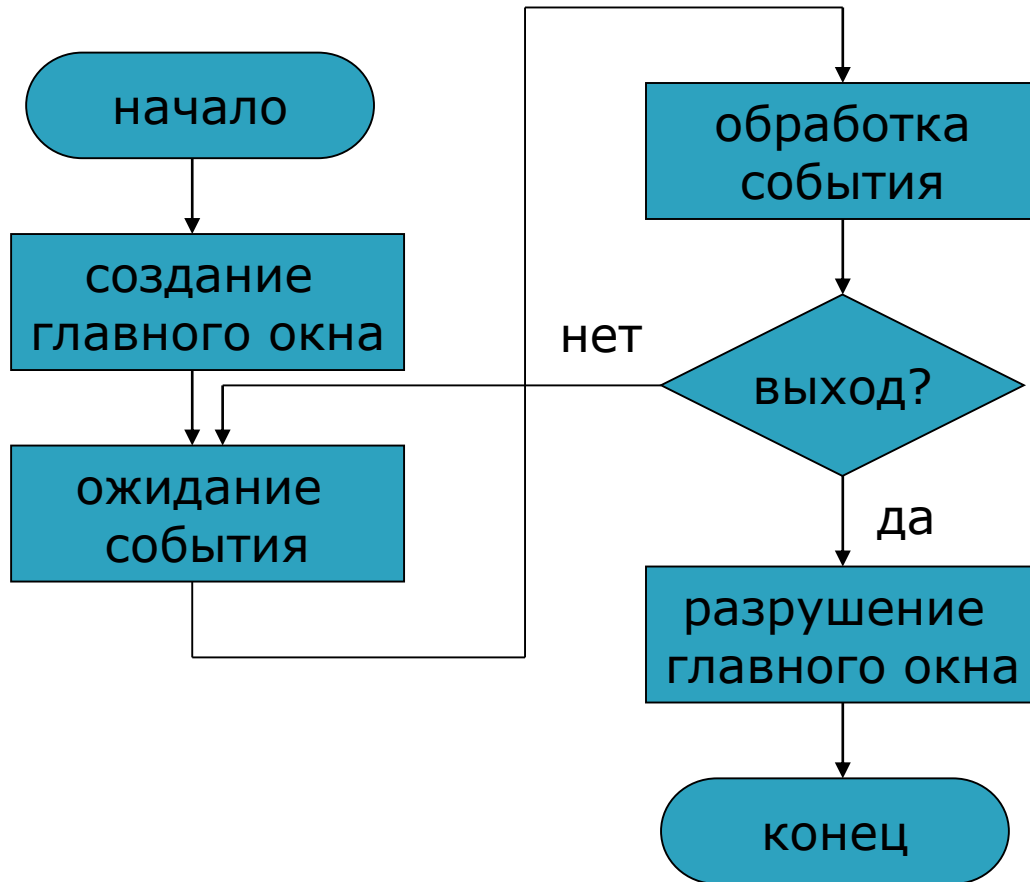
GUI–концепции

- ▶ Программа **отвечает на запросы**, а не действует активно
 - Консольное приложение получает задачу и решает её
 - GUI–приложение функционирует в режиме «вопрос (пользователя) – ответ (программы)»
 - Иногда переключаясь на режим «вопрос (программы) – ответ (пользователя)»

GUI–концепции

- ▶ Программа **отвечает на запросы**, а не действует активно
- ▶ Механизм **событий** (сообщений, сигналов) для получения информации от пользователя (окружения)

Схема работы GUI-приложения



GUI–концепции

- ▶ Механизм **компонентов** (элементов, элементов управления) для представления графической информации

GUI–концепции

- ▶ Механизм **компонентов** (элементов, элементов управления) для представления графической информации
 - Компонент ~ прямоугольник с содержимым

GUI–концепции

- ▶ Механизм **компонентов** (элементов, элементов управления) для представления графической информации
 - Компонент ~ прямоугольник с содержимым
 - Контейнер = содержит другие компоненты

GUI–концепции

- ▶ **Механизм компонентов** (элементов, элементов управления) для представления графической информации
 - Компонент ~ прямоугольник с содержимым
 - Контейнер = содержит другие компоненты
 - Менеджер размещения: управляет размещением компонентов в контейнере

GUI–концепции

- ▶ Механизм **компонентов** (элементов, элементов управления) для представления графической информации
- ▶ Механизм перерисовки через события

GUI–концепции

- ▶ Механизм **компонентов** (элементов, элементов управления) для представления графической информации
- ▶ Механизм перерисовки через события
 - Как по команде от ОС

GUI–концепции

- ▶ Механизм **компонентов** (элементов, элементов управления) для представления графической информации
- ▶ Механизм перерисовки через события
 - Как по команде от ОС
 - Так и по команде от программы

GUI–концепции

- ▶ Механизм **компонентов** (элементов, элементов управления) для представления графической информации
- ▶ Механизм перерисовки через события
 - Как по команде от ОС
 - Так и по команде от программы
 - Набор графических примитивов для отрисовки

GUI–концепции

- ▶ Механизм **компонентов** (элементов, элементов управления) для представления графической информации
- ▶ Механизм перерисовки через события
- ▶ Редактор форм

GUI–концепции

- ▶ Механизм **компонентов** (элементов, элементов управления) для представления графической информации
- ▶ Механизм перерисовки через события
- ▶ Редактор форм
 - Предназначен для интерактивного описания GUI–компонентов
 - Поддерживается IDE и/или плагинами к ней

Организация программы

- ▶ Модель / представление

Организация программы

- ▶ Модель / представление
 - Модель (ядро, бэк-энд): описывает внутреннюю логику, общается с внешним миром через API классов / функций

Организация программы

- ▶ Модель / представление
 - Модель (ядро, бэк-энд): описывает внутреннюю логику, общается с внешним миром через API классов / функций
 - Представление (оболочка, фронт-энд): описывает механизмы взаимодействия с пользователем

Виды событий

- ▶ События обеспечивают связь...
 - с периферийными устройствами (мышь, клавиатура)
 - с изменением состояния одного из компонентов приложения
 - с изменением состояния рабочего стола
 - с изменением состояния нитей, таймеров и других составляющих приложения
 - ...

GUI для Java

- ▶ Библиотеки AWT + Swing (= JFC)

GUI для Java

- ▶ Библиотеки AWT + Swing (= JFC)
 - AWT (1995 год)
 - Поддержка GUI через peer-интерфейсы
 - Метод на Java вызывает «родную» (native) функцию ОС для работы с GUI

GUI для Java

- ▶ Библиотеки AWT + Swing (= JFC)
 - AWT (1995 год)
 - Поддержка GUI через peer-интерфейсы
 - Метод на Java вызывает «родную» (native) функцию ОС для работы с GUI
 - Swing (1998 год)
 - «Тяжёлые» компоненты ~ как в AWT
 - «Лёгкие» компоненты = напрямую не связаны с API операционной системы
 - Кросс-платформенная архитектура = приложение одинаково выглядит везде

GUI для Java

- ▶ Библиотеки AWT + Swing (= JFC)
- ▶ Библиотека JavaFX

GUI для Java

- ▶ Библиотеки AWT + Swing (= JFC)
- ▶ Библиотека JavaFX
 - Создана в 2007–2008
 - Вошла в JRE / JDK в 2014
 - Не только Desktop, но и Web
 - Поддержка CSS (стилей)
 - Считается более современной и продвинутой

GUI для Java

- ▶ Библиотеки AWT + Swing (= JFC)
- ▶ Библиотека JavaFX
- ▶ Android SDK
 - Разработка под ОС Android, 2009 год

GUI для Java

- ▶ Библиотеки AWT + Swing (= JFC)
- ▶ Библиотека JavaFX
- ▶ Android SDK
- ▶ Другие
 - SWT (Eclipse, 2003)
 - Qt Jambi (open-source)

GUI для Kotlin/JVM

- ▶ Всё, что есть для Java +
 - tornadofx
 - DSL для JavaFX

GUI для Kotlin/JVM

- ▶ Всё, что есть для Java +
 - tornadofx
 - DSL для JavaFX
 - anko
 - DSL для Android SDK

GUI для Kotlin/JVM

- ▶ Всё, что есть для Java +
 - tornadofx
 - DSL для JavaFX
 - anko
 - DSL для Android SDK
- ▶ DSL = Domain Specific Language
(проблемно-ориентированный язык)

Простейшее AWT-приложение

- ▶ Все, что требуется сделать – создать окно *Frame*, задать его размеры *setSize*, отобразить его на экране *setVisible* и создать обработчик закрытия окна *addWindowListener*
- ▶ Логика создания нити обработки событий
защита внутри библиотеки

Простейшее AWT-приложение (part3.simple.hello.awt)

```
public class MainFrame extends Frame {  
    MainFrame(String s) {  
        super(s);  
        setSize(400, 400);  
        setVisible(true);  
        addWindowListener(new WindowAdapter() {  
            public void windowClosing(WindowEvent e) {  
                System.exit(0);  
            }  
        });  
    }  
    public static void main(String[] args) {  
        new MainFrame("Приложение AWT");  
    }  
}
```

Простейшее Swing-приложение

- ▶ Все делается примерно аналогично
- ▶ Вместо класса *Frame* (AWT) используется класс *JFrame* (Swing)

Простейшее Swing-приложение (part3.simple.hello.swing)

```
public class MainFrame extends JFrame {  
    MainFrame(String s) {  
        super(s);  
        setSize(400, 400);  
        setVisible(true);  
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
    }  
    public static void main(String[] args) {  
        SwingUtilities.invokeLater(new Runnable() {  
            public void run() {  
                new MainFrame("Приложение Swing");  
            }  
        });  
    }  
}
```

Вызов конструктора фрейма

- ▶ Создателями Swing рекомендуется вызывать конструктор главного фрейма в нити обработки сообщений
- ▶ Для этой цели служит метод `SwingUtilities.invokeLater(runnable);`
- ▶ `Runnable` – интерфейс "запускаемый" с одним методом `run`

Простейшее Swing-приложение (part3.simple.hello.swing)

```
public class MainFrame extends JFrame {  
    MainFrame(String s) {  
        super(s);  
        setSize(400, 400);  
        setVisible(true);  
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
    }  
    public static void main(String[] args) {  
        SwingUtilities.invokeLater(  
            () -> new MainFrame("Простое окно Swing")  
        );  
    }  
}
```

Простейшее JavaFX-приложение (part3.simple.hello.javaafx)

```
class HelloView : View("Простое окно JavaFX") {  
    override val root = BorderPane()  
}
```

```
class HelloApp : App(HelloView::class)
```

```
fun main(args: Array<String>) {  
    Application.launch(  
        HelloApp::class.java, *args)  
}
```

AWT-компоненты (part3.simple.components.awt)

- ▶ Label – текстовая строка
- ▶ Button – клавиша
- ▶ Checkbox – пункт выбора
- ▶ Choice – выпадающий список
- ▶ List – постоянный список

Swing-компоненты (part3.simple.components.swing)

- ▶ JLabel – текстовая строка
- ▶ JButton – клавиша
- ▶ JCheckBox – пункт выбора
- ▶ JComboBox – выпадающий список
- ▶ JList – постоянный список

JavaFX–компоненты (part3.simple.components.javafx)

- ▶ Label
- ▶ Button
- ▶ CheckBox
- ▶ ComboBox

Примеры с компонентами AWT/Swing / JavaFX

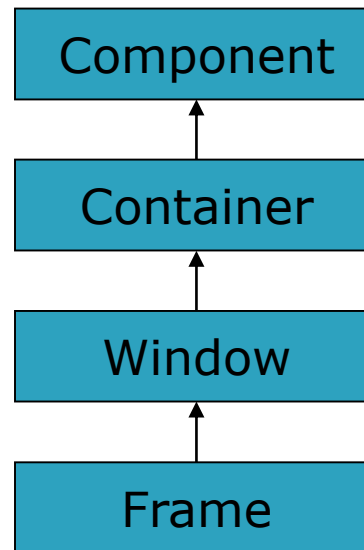
- ▶ Обратите внимание на отличия
компонентов AWT и Swing

Контейнеры

- ▶ Контейнер – это компонент, который может содержать другие компоненты
 - окно
 - панель
 - диалог
 - ...

Иерархия компонентов

- ▶ Контейнер является компонентом (Container extends Component)
- ▶ Окно является контейнером (Window extends Container)
- ▶ Фрейм является окном (с рамкой) (Frame extends Window)



Принципы отображения графики

- ▶ Отображение содержимого окна (контейнера) осуществляется в методе *paint*
 - И только в нём!

Принципы отображения графики

- ▶ Отображение содержимого окна (контейнера) осуществляется в методе *paint*
- ▶ Вывод графических примитивов осуществляется через классы управления графическим контекстом (Graphics, Graphics2D)

Метод paint

- ▶ В результате действий пользователя окно может полностью или частично скрываться и затем открываться снова
- ▶ Поэтому окно должно иметь возможность отобразить свое содержимое в любой момент работы приложения

Метод paint

- ▶ Вызов из ОС
 - В тот момент, когда графическая оболочка считает, что окно должно быть перерисовано, вызывается его метод *paint*

Метод paint

▶ Вызов из ОС

- В тот момент, когда графическая оболочка считает, что окно должно быть перерисовано, вызывается его метод *paint*

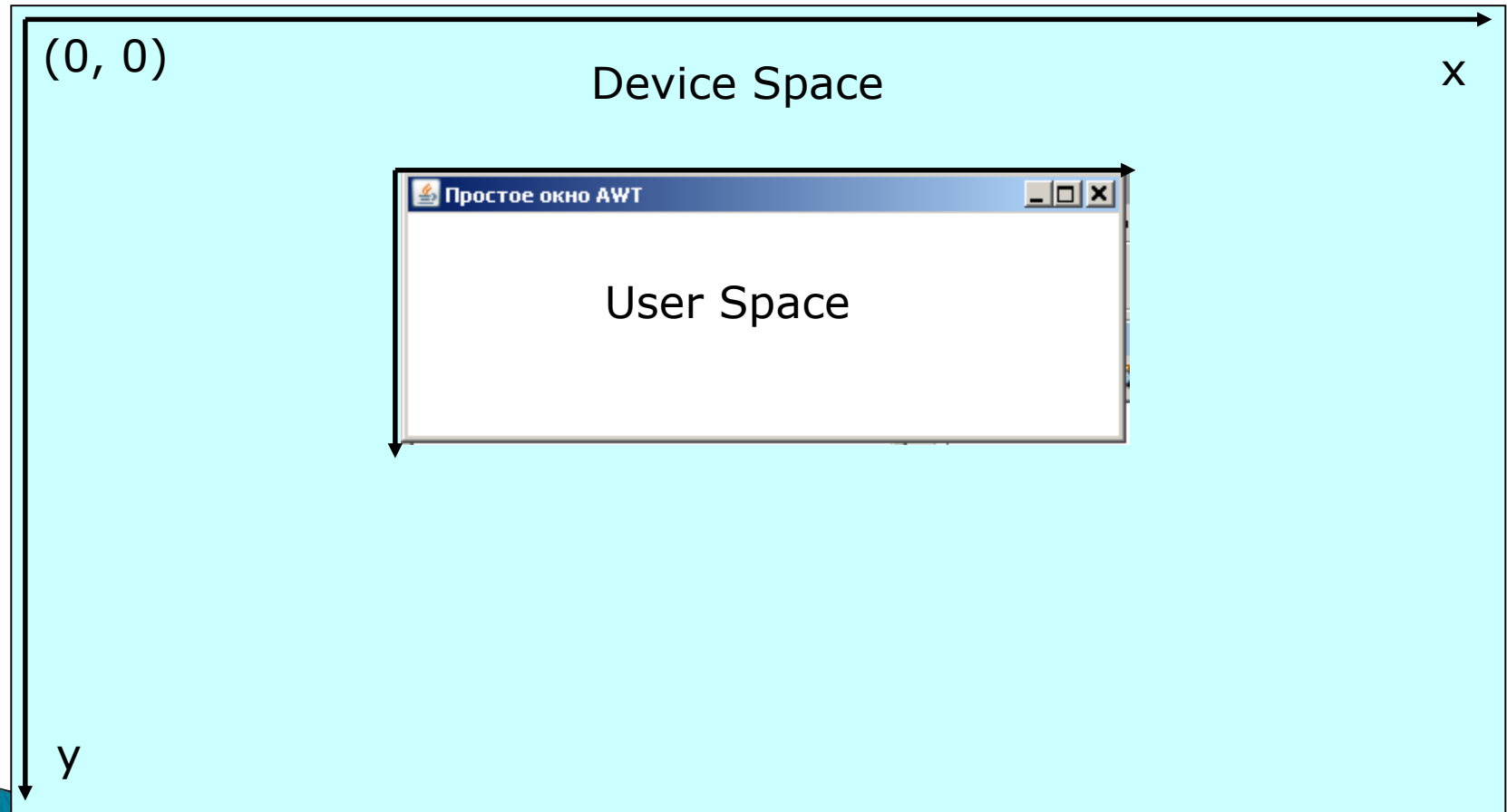
▶ Вызов из программы

- Программа может сама вызвать перерисовку своего окна целиком или частично, вызвав метод *repaint*
 - `repaint();` // или
 - `repaint(x, y, width, height);`

Метод paint

- ▶ В этом методе окно всегда должно быть перерисовано **полностью**
- ▶ Фактически, отрисовка осуществляется не на экране, а в памяти
- ▶ После этого область, которую необходимо перерисовать, копируется из памяти на экран

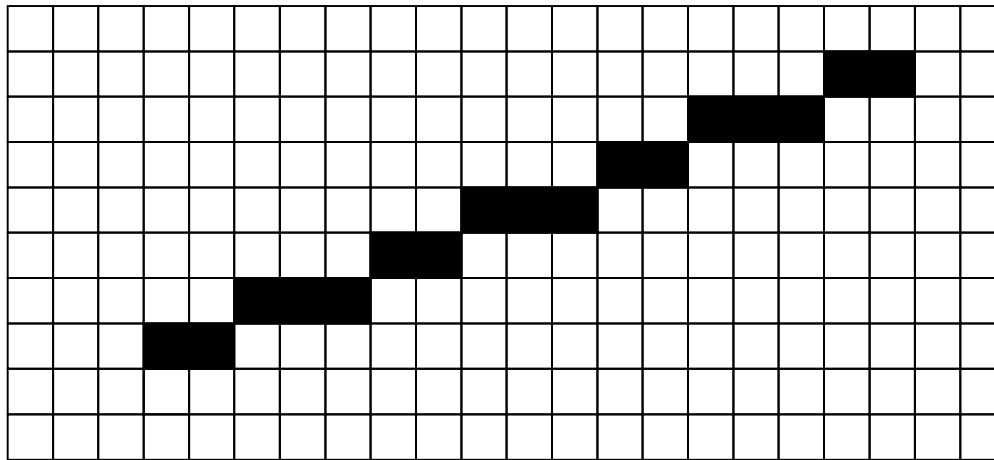
Координатная сетка



Пиксели

- ▶ Пиксел (pixel) – атомарный элемент графического изображения, квадрат с маленькими сторонами
- ▶ Каждый пиксел может иметь свой цвет
- ▶ Пикселы имеют целочисленные координаты:
 $0 \leq x < X_{\max}$, $0 \leq y < Y_{\max}$
- ▶ Все элементы изображения складываются из пикселей

Пример – построение линии



Цвета

- ▶ Цвет пикселя задается интенсивностями красного, зеленого и синего цвета, в диапазоне от 0 до 255
 - `int r=0, g=255, b=255;`
 - `Color color = new Color(r, g, b);`

Прозрачность (alpha)

- ▶ Для цвета может быть задана прозрачность (alpha) в диапазоне от 0 до 255. 255 – непрозрачный цвет, 0 – полностью прозрачный цвет (не будет виден)
 - `Color color = new Color(r, g, b, alpha);`
 - По умолчанию цвета непрозрачны

Выбор цвета

- ▶ Текущий цвет, используемый при рисовании, выбирается отдельным методом *setColor* класса *Graphics*
- ▶ Кроме этого, может быть выбран цвет фона окна, для этого используется метод *setBackground* класса *Component*

Основные примитивы

- ▶ `drawLine(x, y, width, height)` – рисование линии
- ▶ `drawRect(x, y, width, height)` – рисование прямоугольника
- ▶ `drawOval(x, y, width, height)` – рисование овала
- ▶ `drawArc(x, y, width, height, start, length)` – рисование дуги

Основные примитивы

- ▶ `fillRect` – закрашенный прямоугольник
- ▶ `fillRoundRect` – закрашенный закругленный прямоугольник
- ▶ `fillOval` – закрашенный овал

Вывод текста

- ▶ Необходимо создать шрифт
Font font = new Font (
"Serif", Font.ITALIC, 24);
- ▶ Затем его выбрать *setFont(font)*
- ▶ Затем вызвать один из методов отображения текста, например, *drawString(string, x, y);*

Пример с примитивами AWT и Swing

- ▶ См.
 - `part3.simple.primitives.awt`
 - `part3.simple.primitives.swing`

Пример с примитивами JavaFX

- ▶ См. `part3.simple.primitives.javafx`

Устройство фрейма Swing

- ▶ Фрейм Swing, в отличие от фрейма AWT, содержит внутри себя еще один контейнер – так называемый `contentPane`



Устройство фрейма Swing

- ▶ Все компоненты, которые добавляются во фрейм Swing, на самом деле добавляются в его `contentPane`



Перерисовка фрейма Swing

- ▶ Если метод *paint* фрейма Swing переопределен, то перерисовка осуществляется на основном фрейме
- ▶ Попытка изменить фон вызовом метода *setBackground* также изменяет фон основного фрейма
- ▶ В некоторых случаях содержимое *contentPane* затирает собой на экране содержимое основного фрейма

Перерисовка фрейма Swing

- ▶ Метод `paint` для контейнера `ContentPane` переопределить мы не можем
- ▶ Но можно добавить во фрейм еще один контейнер – `JPanel` – и назначить `ContentPane` (см. `part3.simple.panel`)



Добавление панели во фрейм

```
MainFrame(String s) {  
    super(s);  
    setSize(600, 400);  
    JPanel panel = new JPanel();  
    panel.setBackground(  
        Color.CYAN);  
    setContentPane(panel);  
    setVisible(true);  
    setDefaultCloseOperation(  
        JFrame.EXIT_ON_CLOSE);  
}
```

Расширение класса «панель»

```
public class MainPanel extends JPanel {  
    public MainPanel() {  
        super();  
    }  
    public void paint(Graphics g) {  
        super.paint(g);  
        g.setColor(Color.BLACK);  
        g.drawLine(0, 0, getWidth() - 1,  
                   getHeight() - 1);  
        g.drawLine(getWidth() - 1, 0, 0,  
                   getHeight() - 1);  
    }  
}
```

Итоги

- ▶ Рассмотрены
 - Основные концепции GUI
- ▶ Далее
 - Редактор форм
 - Создание изображений
 - ...