

GUI

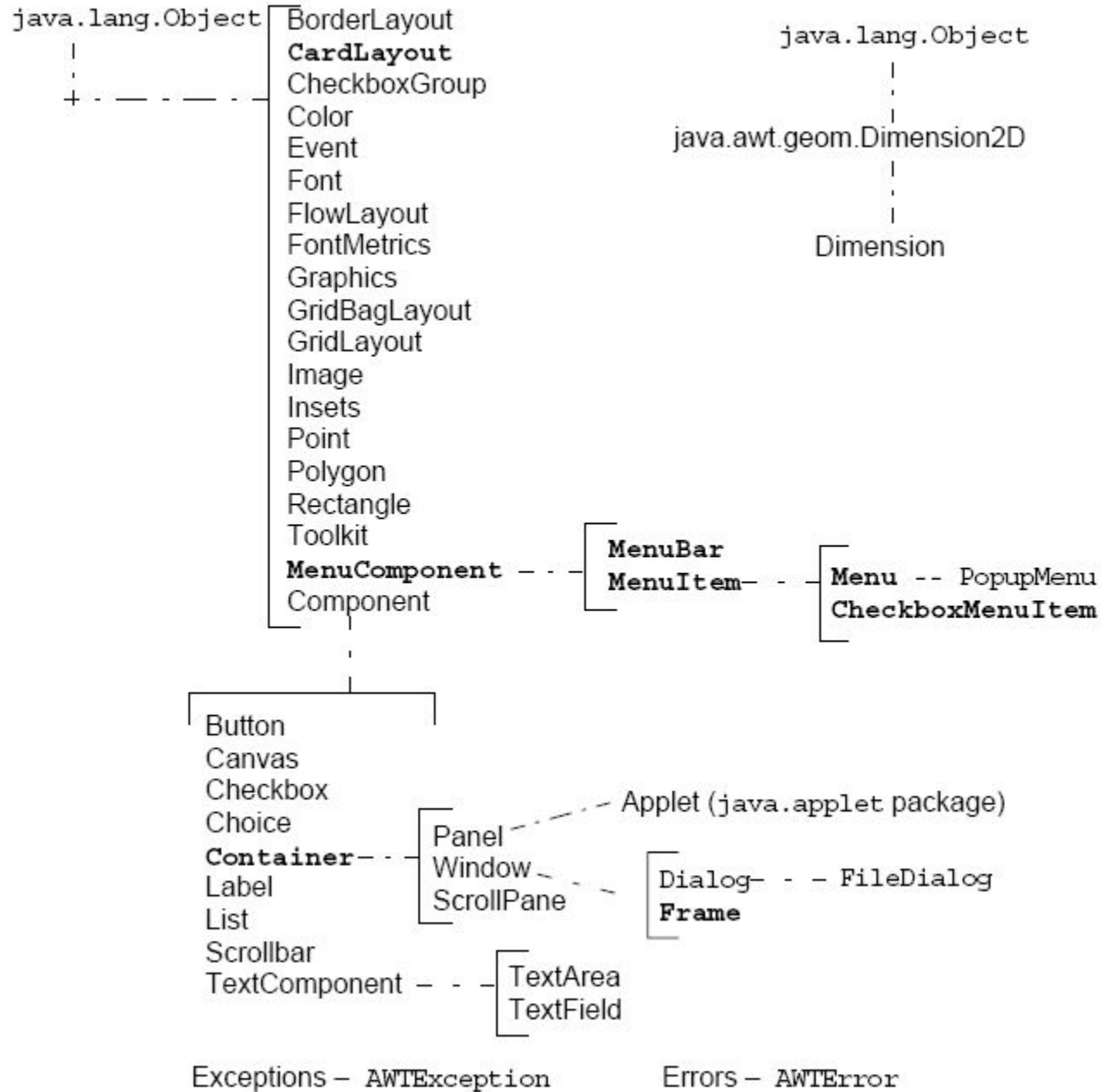
GUI

- AWT (Abstract Windows Toolkit)
- Swing

AWT / Swing

- Native кодирование
- Используются платформо-зависимые компоненты
- Внешний вид и функциональность определяется платформой
- Чистая Java
- Lightweight
- Более широкие возможности
- Полная независимость от платформы

AWT



Компонент

Элемент GUI, отображающийся на экране, осуществляющий взаимодействие с пользователем.

- Component
- MenuComponent

Контейнер

Компонент, который может содержать другие компоненты

- Panel
 - Applet
- Window
 - Dialog
 - Frame
- ScrollPane

Color

- Класс, отвечающий за цвет

Использование:

1) Статические цвета – `Color.RED`,
`Color.BLACK` и т.д.

2) Создание цвета в RGB

```
Color c = new Color(255,0,0);
```

Frame

- Наследник `Window`
- Имеет заголовок
- Можно изменять размер мышью
- Изначально невидим
 - Используйте `setVisible(true)`
- По умолчанию использует менеджер размещения `BorderLayout`
 - Для изменения используйте `setLayout`


```
import java.awt.*;

public class FrameExample {
    private Frame f;

    public FrameExample() {
        f = new Frame("Hello Out There!");
    }

    public void launchFrame() {
        f.setSize(170,170);
        f.setBackground(Color.blue);
        f.setVisible(true);
    }

    public static void main(String args[]) {
        FrameExample guiWindow = new FrameExample();
        guiWindow.launchFrame();
    }
}
```

Panel

- Позволяет размещать компоненты и другие панели
- Вложенные панели могут иметь различные менеджеры размещения

```
import java.awt.*;

public class FrameWithPanel {
    private Frame f;
    private Panel pan;

    public FrameWithPanel(String title) {
        f = new Frame(title);
        pan = new Panel();
    }

    public void launchFrame() {
        f.setSize(200,200);
        f.setBackground(Color.blue);
        f.setLayout(null); // Попробуйте закомментировать эту строку
        pan.setSize(100,100);
        pan.setBackground(Color.yellow);
        f.add(pan);
        f.setVisible(true);
    }

    public static void main(String args[]) {
        FrameWithPanel guiWindow = new FrameWithPanel("Frame with Panel");
        guiWindow.launchFrame();
    }
}
```

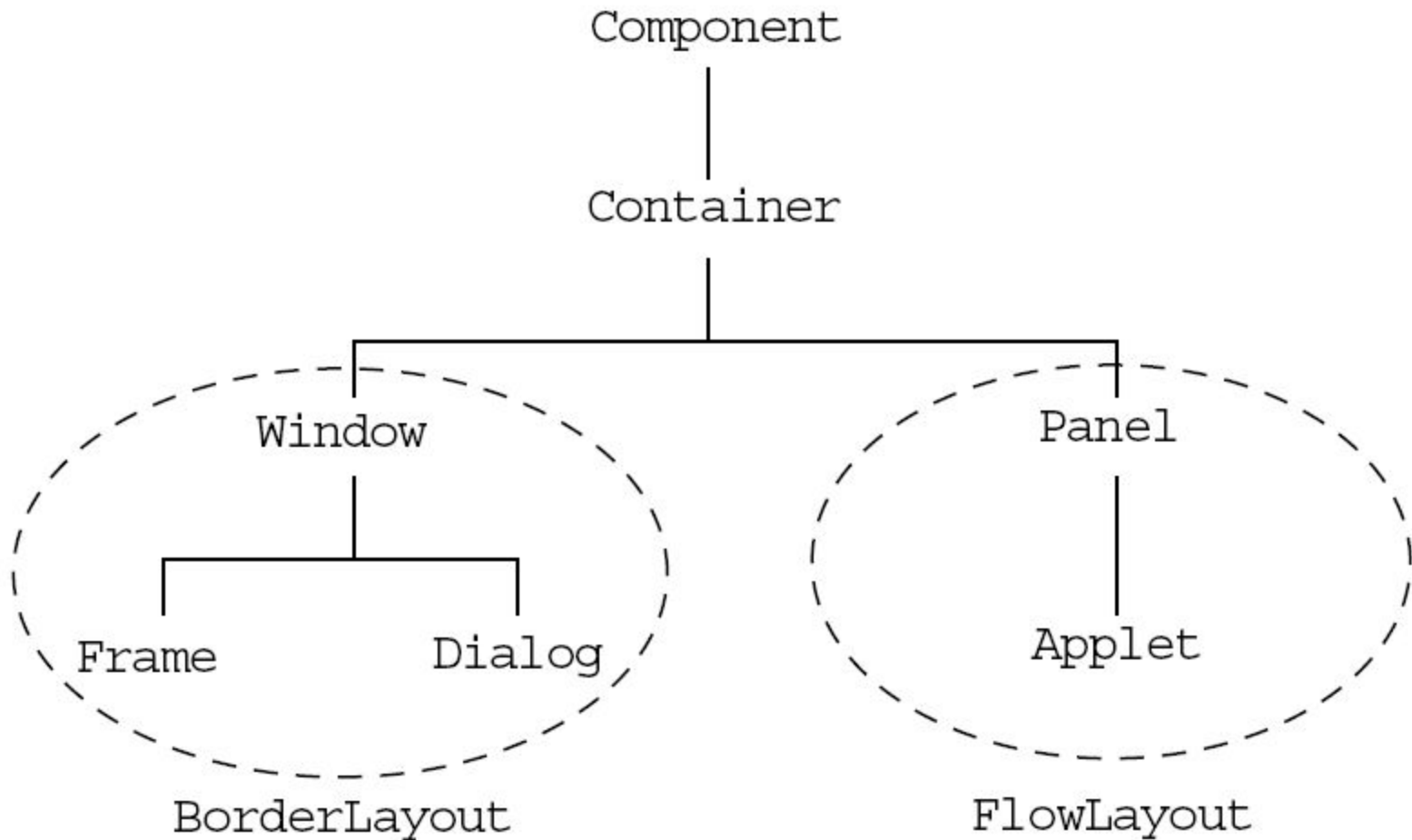
Размещение компонентов

- Непосредственное указание положения и размеров
 - Используйте для контейнера **setLayout(null)**, а для компонентов **setLocation(...)**, **setSize(...)** или **setBounds(...)**
- Использование менеджеров размещения

Менеджеры размещения (Layout Managers)

- `FlowLayout`
- `BorderLayout`
- `GridLayout`
- `CardLayout`
- `GridBagLayout`

Менеджеры по умолчанию



FlowLayout

- Используется по умолчанию для Panel
- Добавляет компоненты слева-направо, сверху-вниз
- По умолчанию компоненты выравниваются по центру
- Используются предпочтительные размеры компонентов

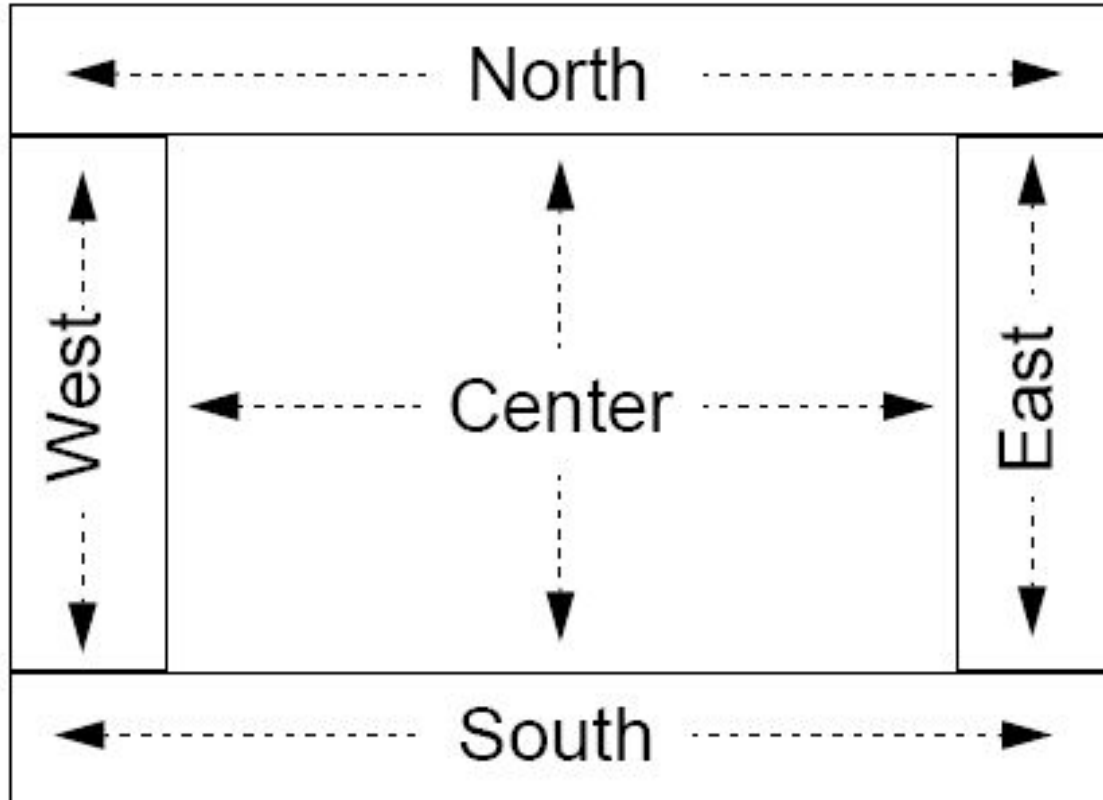
```
import java.awt.*;
public class LayoutExample {
    private Frame f;
    private Button b1;
    private Button b2;

    public LayoutExample() {
        f = new Frame("GUI example");
        b1 = new Button("Press Me");
        b2 = new Button("Don't press Me");
    }

    public void launchFrame() {
        f.setLayout(new BorderLayout());
        f.add(b1);
        f.add(b2);
        f.pack(); // Попробуйте закомментировать эту строку
        f.setVisible(true);
    }

    public static void main(String args[]) {
        LayoutExample guiWindow = new LayoutExample();
        guiWindow.launchFrame();
    }
}
```


BorderLayout



BorderLayout

- Используется по умолчанию для Frame
- Компоненты добавляются в указанный регион (**North**, **South**, **West**, **East**, **Center**)
- Для **North** и **South** используется предпочтительная высота элементов
- Для **West** и **East** используется предпочтительная ширина элементов
- Оставшееся место – для **Center**

```
import java.awt.*;
public class BorderExample {
    private Frame f;
    private Button bn, bs, bw, be, bc;
    public BorderExample() {
        f = new Frame("Border Layout");
        bn = new Button("North");
        bs = new Button("South");
        bw = new Button("West");
        be = new Button("East");
        bc = new Button("Center");
    }
    public void launchFrame() {
        f.add(bn, BorderLayout.NORTH);
        f.add(bs, BorderLayout.SOUTH);
        f.add(bw, BorderLayout.WEST);
        f.add(be, BorderLayout.EAST);
        f.add(bc, BorderLayout.CENTER);
        f.setSize(200,200); // Попробуйте заменить на f.pack();
        f.setVisible(true);
    }
    public static void main(String args[]) {
        BorderExample guiWindow2 = new BorderExample();
        guiWindow2.launchFrame();
    }
}
```

GridLayout

- В конструкторе указывается количество строк и столбцов
- Элементы добавляются слева-направо, сверху-вниз
- Все элементы имеют одинаковые размеры

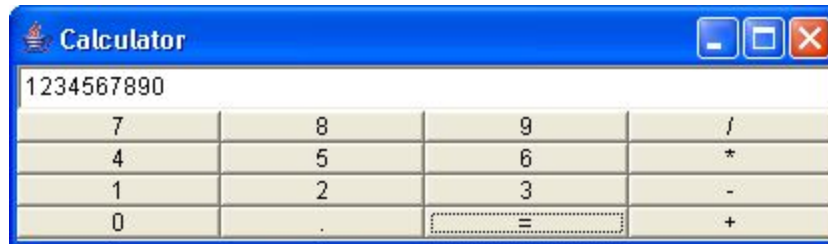
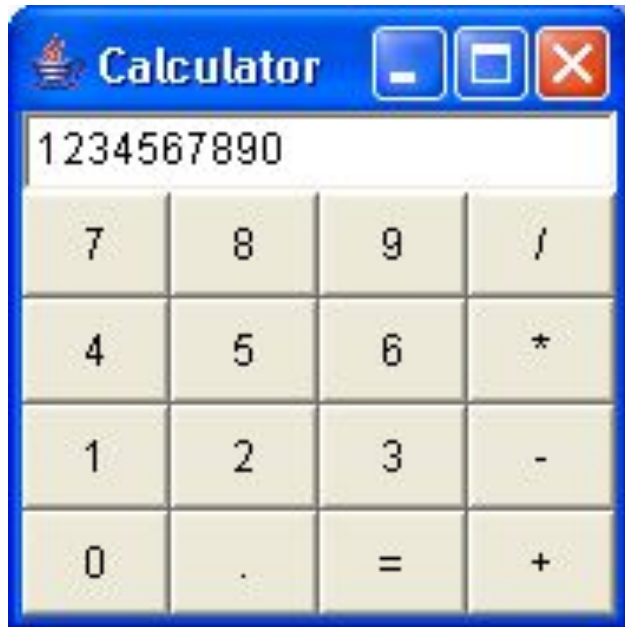
```
import java.awt.*;
public class GridExample {
    private Frame f;
    private Button b1, b2, b3, b4, b5, b6;
    public GridExample() {
        f = new Frame("Grid Example");
        b1 = new Button("1"); b2 = new Button("2");
        b3 = new Button("3"); b4 = new Button("4");
        b5 = new Button("5"); b6 = new Button("6");
    }
    public void launchFrame() {
        f.setLayout (new GridLayout(3,2));
        f.add(b1);    f.add(b2);
        f.add(b3);    f.add(b4);
        f.add(b5);    f.add(b6);
        f.pack();
        f.setVisible(true);
    }
    public static void main(String args[]) {
        GridExample grid = new GridExample();
        grid.launchFrame();
    }
}
```

Комбинирование менеджеров размещения с помощью вложенных панелей

```
import java.awt.*;
public class ComplexLayoutExample {
    private Frame f;
    private Panel p;
    private Button bw, bc, bfile, bhhelp;
    public ComplexLayoutExample() {
        f = new Frame("GUI example 3");
        bw = new Button("West"); bc = new Button("Work space region");
        bfile = new Button("File"); bhhelp = new Button("Help");
    }
    public void launchFrame() {
        f.add(bw, BorderLayout.WEST);
        f.add(bc, BorderLayout.CENTER);
        p = new Panel();
        p.add(bfile);
        p.add(bhhelp);
        f.add(p, BorderLayout.NORTH);
        f.pack();
        f.setVisible(true);
    }
    public static void main(String args[]) {
        ComplexLayoutExample gui = new ComplexLayoutExample();
        gui.launchFrame();
    }
}
```

Задание 1

Написать программу – графический интерфейс калькулятора (пока без реакции на действия пользователя).



Задание 2

Написать программу – графический интерфейс Chat-клиента (пока без реакции на действия пользователя).



Задание 3

Написать консольную программу, в которой пользователь вводит желаемое количество прямоугольников. После чего выводится окно с данным количеством прямоугольников случайного цвета. Их размещение должно соответствовать области n на n , размер которой зависит от количества прямоугольника.