Абстрактный оконный интерфейс (Abstract Window Toolkit AWT)

В настоящее время большинство java-программ построены на основе пользовательского интерфейса **Swing**. **Swing** предлагает более широкие возможности, чем **AWT**, в отношении создания распространенных компонентов GUI (кнопки, окна, списки и т.д.). Но: Swing построен на базе AWT, многие классы AWT используются в Swing.

Классы **AWT** содержатся в пакете **java.awt**.
AWT определяет окна в соответствии с иерархией классов, которые добавляют функциональность и специфичность на каждом уровне.

Два наиболее часто используемый тип окна наследуются от класса **Frame** (создает стандартное окно приложения).

На вершине иерархии находится класс **Component** - это абстрактный класс, инкапсулирующий все атрибуты визуального компонента. **Component** определяет свыше сотни общедоступных методов, отвечающих за управление событиями, такими как ввод с клавиатуры, работа с мышью, перемещение и изменение размеров окна, а также перерисовка, запоминание текущих цветов фона и переднего плана, а также текущего шрифта для печати.

Container - подкласс **Component**, имеет дополнительные методы, позволяющие вкладывать в него другие объекты **Component**. Другие объекты **Container** также могут находиться внутри **Container** (поскольку они сами являются экземплярами Component). Это позволяет строить многоуровневые системы вложенности. **Container** отвечает за расположение любых компонентов, которые он содержит.

Класс **Window** - создает окно верхнего уровня. Окно верхнего уровня не содержится внутри другого объекта, а располагается на рабочем столе. Обычно используется **Frame**, подкласс **Window**.

Класс **Frame** представляет то, что обычно воспринимается как «окно», имеет линейку заголовка, линейку меню, границы и элементы управления размером. Также окно **Frame** создается для приложения как обычное окно.

Canvas - инкапсулирует пустое окно для рисования.

Пакет java.awt

Классы **Color** **Font** **Component** и др.

Button **Label** **Container** **Canvas** **Scrollbar** и др.

Window

Frame

Checkbox, Choice, List, TextComponent

Методы класса Component (напр., setBounds(…)) доступны для кнопок, меток и др. подклассов (компонентов).

Методы класса Container позволяют включать компоненты в окна, фреймы и апплеты, менять размеры окон и апплетов.

Класс **Frame** из пакета **java.awt**

Иерархия классов:

Component → Container → Window →Frame

(все классы обеспечивают поддержку AWT)

• Класс **Window**

Создает окно верхнего уровня на рабочем столе. Он расширяется классом Frame, который и представляет интерфейсное окно, окно с меню, обрамлением, необходимое для создания графического приложения с его компонентами.

•Класс **Frame**

Инкапсулирует полноценное окно, имеющее строку заголовка, строку меню, обрамление и углы, изменяющие размеры окна.
Для создания окна **Frame** существуют два конструктора:

Frame ( );

Frame (String name);

Для установления размера фрейма существуют методы:
void setSize (int Width, int Height);

void setSize (Dimension size), где Dimension – класс, содержащий поля width и height.

Метод, позволяющий сделать окно видимым:
void setVisible (boolean visibleFlag);

String getTitle( ); – получить заголовок окна;
void setTitle (String); – установить заголовок окна;
void setResizable (boolean); **-** разрешить изменение размеров окна;
boolean isResizable(); **-** возвратить true, если размер окна можно изменять, иначе false.

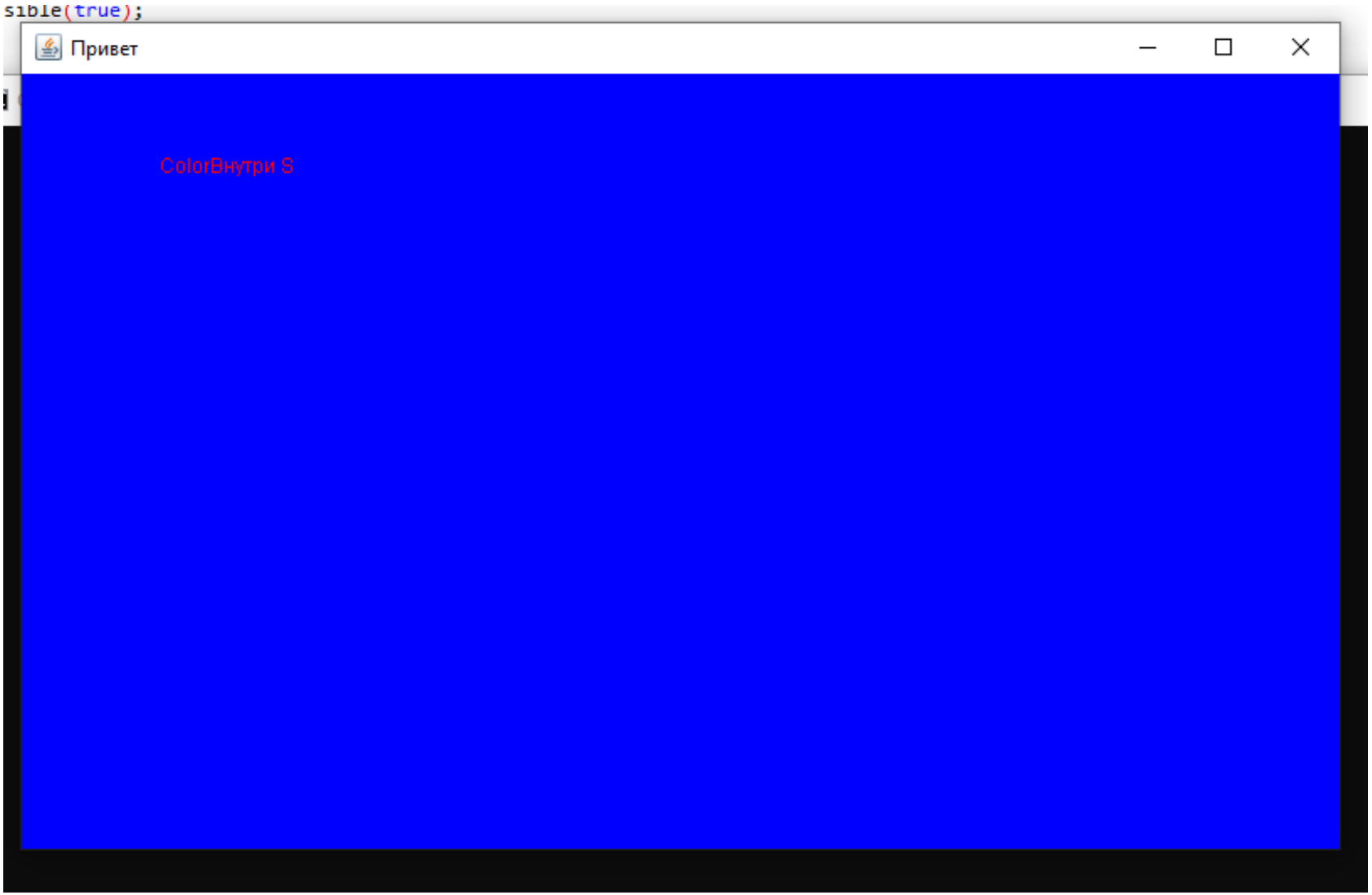
void paint(Graphics g) – вызывается для перерисовки окна, т.е. при сворачивании или перекрытии окна, **Graphics g** описывает графическую среду окна;

public void repaint() **-** вызывает paint, если нужно обновить содержимое окна (несколько разных вариантов вызова).

Создание окна

```
import java.awt.*;
import java.awt.event.*;

public class SColor extends Frame implements Runnable{
   String msg;
   Thread t=null; //потоковая переменная


public SColor(String m) {    // конструктор
    super(m);                  // обращение к конструктору суперкласса
    setSize(800,500);                  // размеры окна
    setBackground(Color.blue);     //цвет фона
    setForeground(Color.red);       //цвет переднего плана
    addWindowListener(new WindowAdapter ( ){
            public void windowClosing(WindowEvent e)  {
                                                System.exit(0);   }   }    );
    msg="Внутри SColor";
    t = new Thread(this,"поток");    //создание потока
    t.start();
    setVisible(true);
 }
```

```
public void run( ){              //действия, которые выполняет поток t
   char ch;
for ( ; ; ) {       //в бесконечном цикле
   try{
      repaint();                                   //перерисовываем (вызов paint())
      Thread.sleep(250);                           //приостанавливаем поток
      ch=msg.charAt(0);                            //получаем символ на позиции 0
      msg=msg.substring(1,msg.length());   //символы с 1 до посл.
      msg+=ch;          }                          //нулевой символ в конец
  catch(InterruptedException e) {    }
 }  }

public void paint(Graphics g) {
      g.drawString(msg, 90, 90);  }    //вывод строки в окно

public static void main(String args []) {
      new SColor("Привет");
   }
}
```


sible(true);
Привет
ColorВнутри S

Класс **Color** пакет **java.awt**

Константы: **Color.black, Color.blue, Color.cyan, Color.green, Color.red, Color.orange** и др.

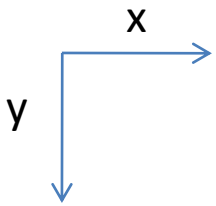
Конструктор: **Color(int red, int green, int blue)**

red, green и **blue** принимают значения от 0 до 255

```
Graphics g;                                  // графическая переменная
Color c = new Color(255,100,100);            //создаем цвет
                                  // или  Color c = new Color(Color.red);
g.setColor(c);          // устанавливаем цвет в графич. перем.
g.drawString("Привет!", 30,30);  // выводим строку цвета c
```

Класс Graphics (пакет java.awt)

Графический контекст апплета инкапсулирует класс Graphics (цвет, шрифт, среда отображения). Вся графика рисуется относительно окна. Начальная точка – верхний левый угол (0,0).



Графический контекст:

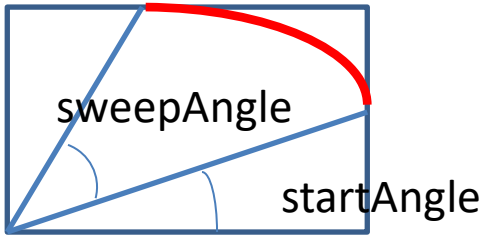
1. передается в апплет при вызове paint(), update();
2. возвращается методом getGraphics() (класс Component)

Объекты рисуются и заливаются текущим выбранным цветом (по ум. черный, новый цвет устанавл. перед прорисовкой фигуры).

1. **drawLine(int X, int Y, int endX, int endY)** – линия;
2. **drawRect(int x, int y, int width, int height)** – прямоугольник,
   fillRect(int x, int y, int width, int height) – залитый прямоугольник;
   drawRoundRect(int x, int y, int width, int height, int xDiam, int yDiam)
   fillRoundRect(int x, int y, int width, int height, int xDiam, int yDiam)
   - прямоугольники с закругленными углами (диаметры округления).

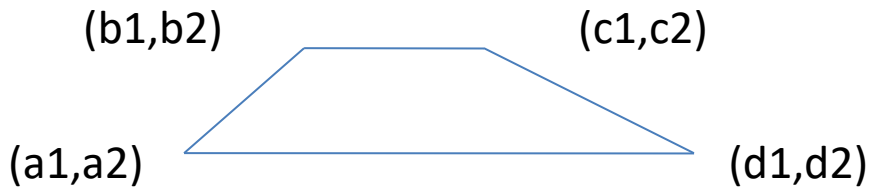
3. **drawOval(int x, int y, int width, int height)** – эллипс/окружность,
fillOval(int x, int y, int width, int height) – залитый эллипс/окружность;
Вписываются в прямоугольник со сторонами width и height.

4. **drawArc(int x, int y, int width, int heigh, int startAngle, int sweepAngle)** –дуга;
Углы задаются в градусах.
startAngle – начальный угол в прямоугольнике.
sweepAngle – угол, который откладывается от startAngle, если с «+», то против часовой стрелки, если с «-» - по часовой стрелке.
fillArc(int x, int y, int width, int heigh, int startAngle, int sweepAngle) – залитая дуга;



5. **drawPolygon(int x[],int y[], int numPoints)** – многоугольник,
fillPolygon(int x[],int y[], int numPoints) – залитый многоугольник;

int x[] = {a1, b1, c1, d1}; int y[] = {a2,b2,c2,d2};



Режимы рисования:

по умолчанию новый вывод в апплет перекрывает старое содержимое.

void setPaintMode() – режим с перекрытием;

void setXORMode(Color xor) – новый объект всегда будет виден, независимо от цвета фона (XOR операция цвета xor и предыдущего цвета).

```
Graphics g;
g.setXORMode(Color.white);
…
g.setPaintMode();
```

Шрифты. Класс Font

Шрифт имеет имя семейства, логическое имя, название гарнитуры, размер.

Конструктор: **Font(String fontName, int fontStyle, int pointSize)**

fontName: Dialog, Monospaced и др.
fontStyle : Font.PLAIN, Font.BOLD, Font.ITALIC

```
Font f = new Font(“Dialog”,Font.PLAIN,12);
setFont(f);
```

```
import java.awt.event.*;
import java.awt.*;

public class Rectangles extends Frame  {
   DrawRect  DR = null;    //поток для рисования прямоугольников
   DrawEll    DE = null;     //поток для рисования эллипсов

public Rectangles(String ms){
   super(ms);
   setSize(800,500);
   addWindowListener (new WindowAdapter ( ) {
         public void windowClosing(WindowEvent e) {
            if (DR != null)  DR = null;
            if (DE != null)   DE = null;
            System.exit(0);   } }  );
  setVisible(true);

if (DR == null) {   //прямоугольники
    DR = new DrawRect(this);
    DR.start();  }
if (DE == null) {  //эллипсы
  DE = new DrawEll(this);
  DE.start(); }      }
```

```
public void paint(Graphics g) {    // закрашиваем фон
      g.setColor(Color.blue);
      g.fillRect(0, 0,800, 500);    }

public static void main(String args []) {
      new Rectangles("Привет");
   }
}
```

```
class DrawRect extends Thread {   //класс рисования прямоугольников
      Graphics g;                  //графическая переменная
public DrawRect(Rectangles M) {     //получаем окно
        g = M.getGraphics();    }          //графический контекст
public void run(   ) {       //действия потока
  while (true) {
       int x, y, width, height;    //координаты, ширина и высота
  int rColor, gColor, bColor;       //переменные цвета
  x = (int)(800* Math.random());  //случайные координаты для
  y = (int)(500* Math.random());  //для вывода прямоугольника
  width = (int) x/2;     // ширина и высота прямоугольника
  height = (int) y/2;
  rColor = (int)(255 * Math.random());    //случайный цвет
  gColor = (int)(255 * Math.random());
  bColor = (int)(255 * Math.random());
  g.setColor(new Color(rColor, gColor, bColor));
  g.fillRect(x, y,height, width);    //рисуем прямоугольник
try { Thread.sleep(50); }             //приостанавливаем поток
catch (InterruptedException e) {  }
} } }
```

```
class DrawEll extends Thread {
            Graphics g;
public DrawEll(Rectangles M) {
          g = M.getGraphics(); }
public void run() {
       while (true) {
           int x, y, width, height;
           int rColor, gColor, bColor;
          x =(int)(800* Math.random());
          y =(int)(500* Math.random());
          width = (int)(x* Math.random()) / 2;
          height = (int)(y* Math.random()) /2;
          rColor = (int)(255 * Math.random());
          gColor = (int)(255 * Math.random());
          bColor = (int)(255 * Math.random());
          g.setColor(new Color(rColor, gColor, bColor));
          g.fillOval(x, y, width, height);
try { Thread.sleep(50); }
catch (InterruptedException e) {  }
} } }
```


Привет
.setColor(new Color(rColor, gColor, bColor));