Farben - Beispiel

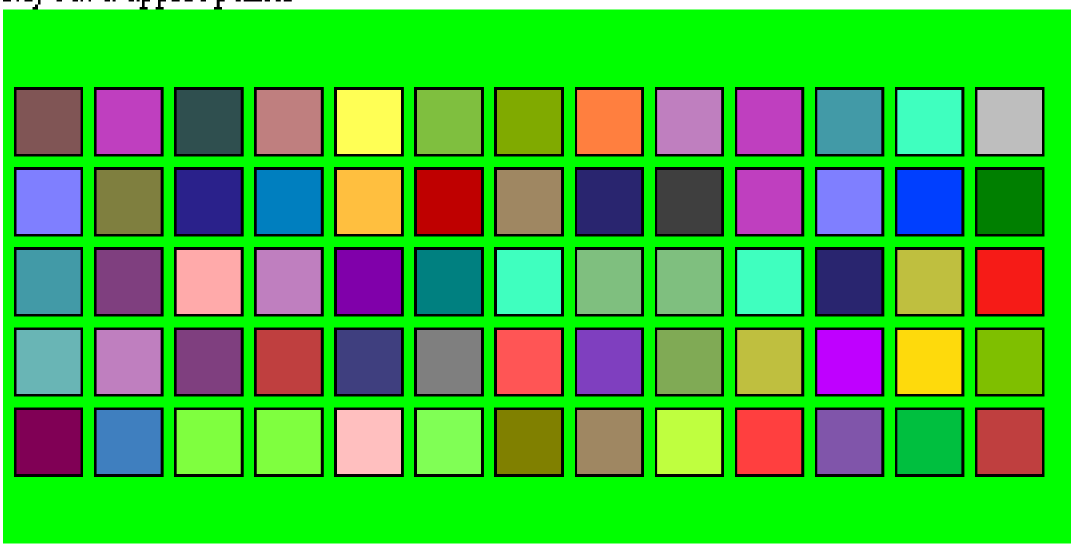
```
private int rd() {
  return (int)(Math.random()*256);   }

public void paint(Graphics g) {
int rval, gval, bval;
for (int j=30; j < (size().height-25); j+=30)
  for (int i=5; i < (size().width-25); i+=30){

     rval = rd(); gval = rd(); bval = rd();

     g.setColor(new Color(rval,gval,bval));
     g.fillRect(i,j,25,25);
     g.setColor(Color.black);
     g.drawRect(i-1,j-1,25,25); };
setBackground(Color.green);
}
```

My Java applet paints



Farben

- Die `Color` Klasse

```
import java.awt.Color;
```

- 24-bit Farben

```
255,0,0       Color.red (final static)
0,255,05      Color.green
0,0,255       Color.blue
255,255,255   Color.white
0,0,0         Color.black
... (Siehe das Programm "Color.java".)
```

- Die `Color` Objekte und die aktuellen Farben

```
public void paint(Graphics g) {
 Color mygray = new Color(128,128,128);
 g.setColor(mygray);
    // Ab jetzt mygray für draw-Methoden.

 g.setColor(Color.green);
    // Ab jetzt green für draw-Methoden.

 g.setBackground(Color.white);
    // Für Hinterground

}
```

Informationen über einen Font

- Die `FontMetrics` Klasse und ihre Objekte

```
import java.awt.Font;
import java.awt.FontMetrics;

public void paint(Graphics g) {
  Font f = new Font(“TimesRoman”,
                          Font.ITALIC,12);
  FontMetrics fm = getFontMetrics(f);
  int w = fm.stringWidth(“Hello”);
  int h = fm.stringHeight();
   ...
}
```

- Beispiel

```
public void paint(Graphics g) {
  ...
  int xstart = (size().width - w) / 2;
     // this.size() ist Größe des Applets
  int xstart = (size().height - h) / 2;

  g.drawSting(“Hello”, xstart, ystart);
}
```

Text und Font

• Die `Font` Klasse und ihre Objekte

```
import java.awt.Font;
...
Font f1 = new Font("Courier",
                              Font.BOLD,24);
Font f2 = new Font("TimesRoman",
                            Font.ITALIC,12);
```

• Der aktuelle Font eines `Graphics`-Objektes

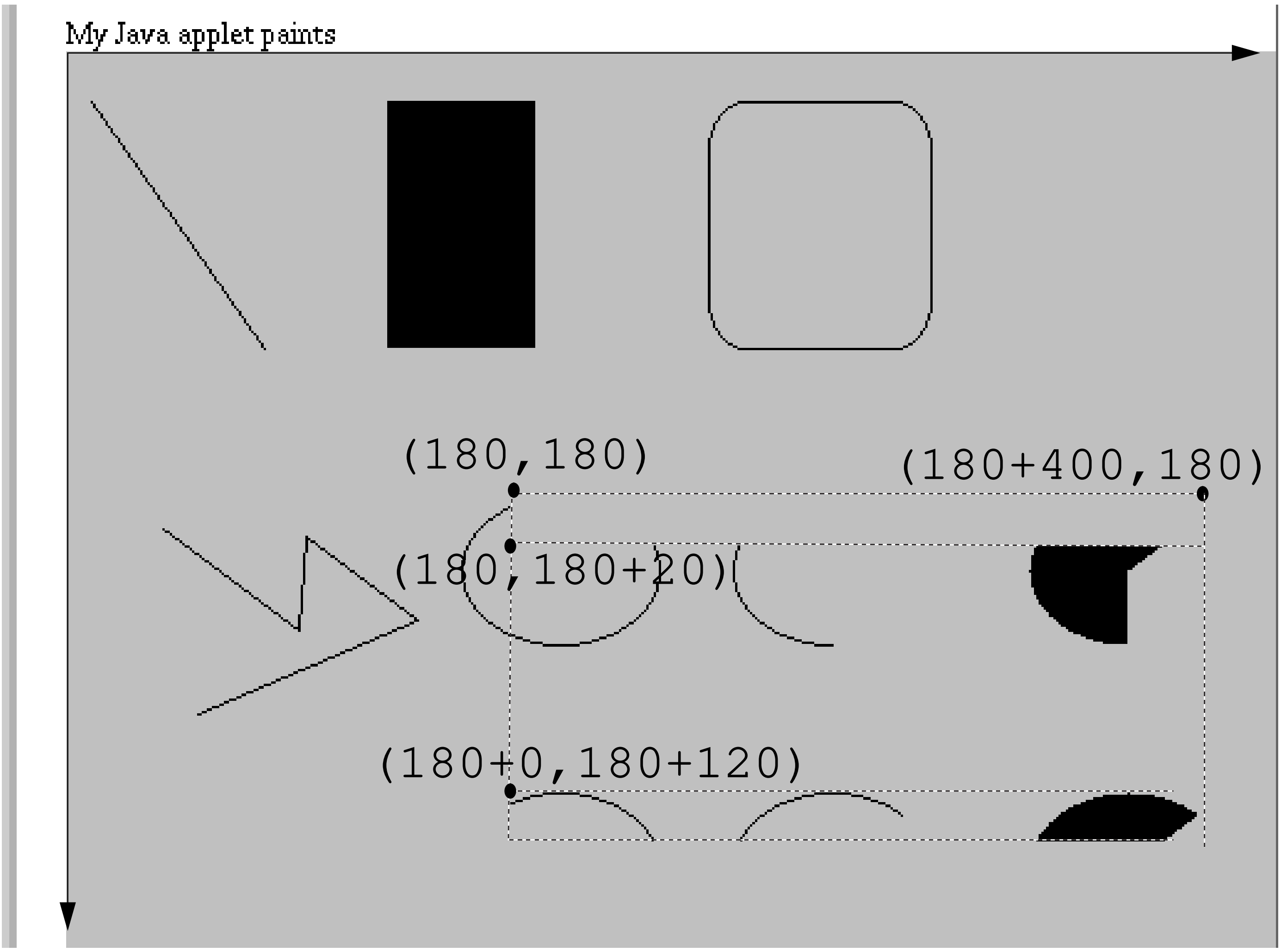
```
public void paint(Graphics g) {
  ...
  g.setFont(f1);
  g.drawString("xxx",10,100);
  g.setFont(f2);
  g.drawString("yyy",10,100);
}
```

• Informationen über einen Font

```
public void paint(Graphics g) {
  ...
  Font f3 = g.getFont();
  String n = f3.getName();
  int s = f3.getSize();
}
```

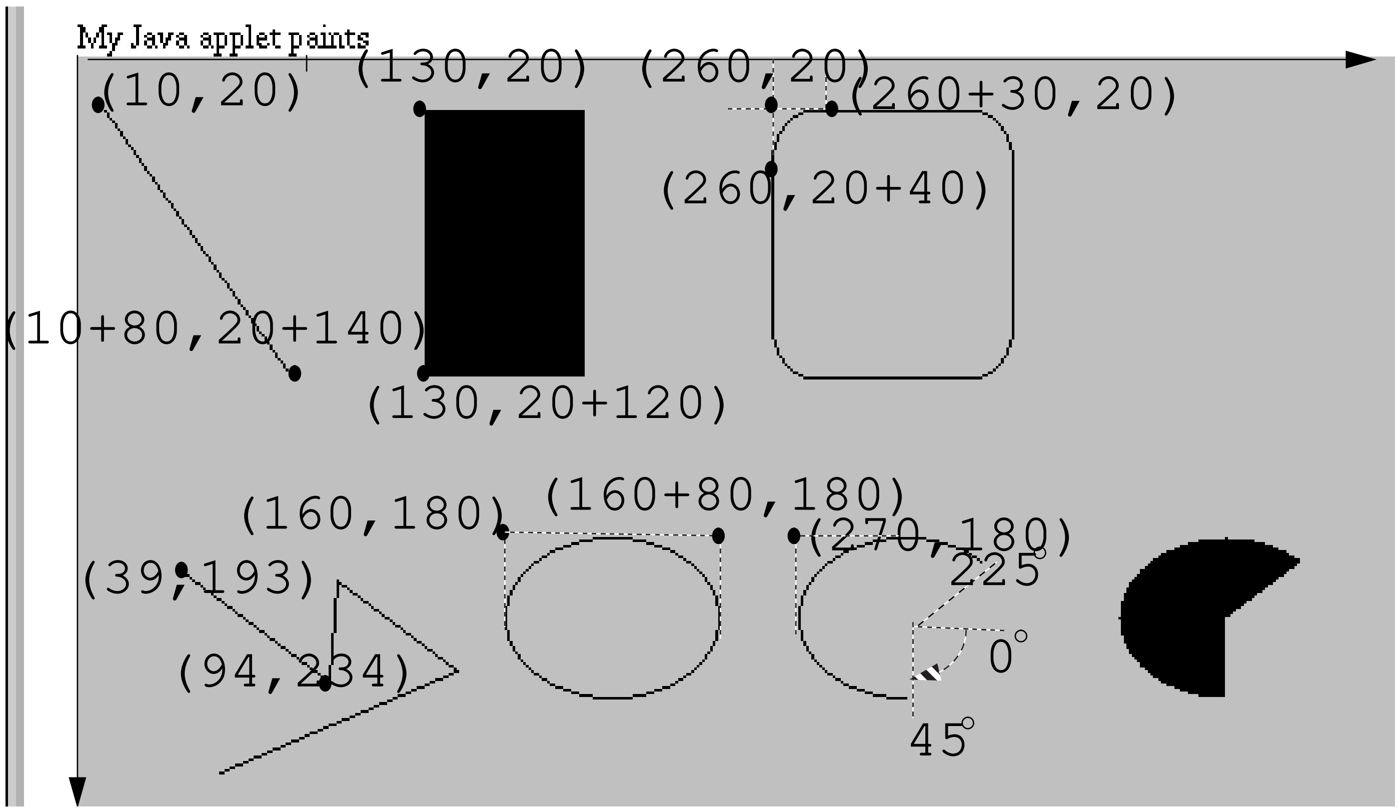
Die `Graphics` Klasse - Fortsetzung

```
public void paint(Graphics g) {
      ... wie früher ...
    g.copyArea(180,180,400,20,0,120);
    g.clearRect(180,180,400,20);
}
```



Die `Graphics` Klasse

```
    public void paint(Graphics g) {
        g.drawLine(10, 20, 80, 120);
        g.fillRect(130, 20, 60, 100);
       g.drawRoundRect(260,20,90,100,30,40);
        int[] exes = {39,94,97,142,53};
        int[] whys = {180+13,180+54,180+16,
                                    180+50,180+88};
        int pts = exes.length;
        g.drawPolygon(exes,whys, pts);
        g.drawOval(160,180,80,60);
        g.drawArc(270,180,80,60,45,225);
        g.fillArc(390,180,80,60,45,225);
}
```



Parameterübergabe für Applets - Fortsetzung

```
public class MoreHelloApplet extends Applet{
    Font theFont;
    int theSize;
    String theName;

    public void init() {
      String s = getParameter("size");
       // Lies den Wert für Parameter "size"
       // null, wenn kein <PARAM> für "size"
      theName = getParameter("name");

      if (s == null) s = "12";
       // Standard-size ist 12
      theSize = Integer.parseInt(s);
      theFont = new Font("TimesRoman",
           Font.BOLD,theSize);
      if (theName == null) theName = "";
      theName = "Hello " + theName + "!";
      }

    public void paint(Graphics g) {
       g.setFont(theFont); // Setze den Font
     g.drawString(theName, 20, theSize + 5);
    }
}
```

Parameterübergabe für Applets

- Teil der HTML-Datei

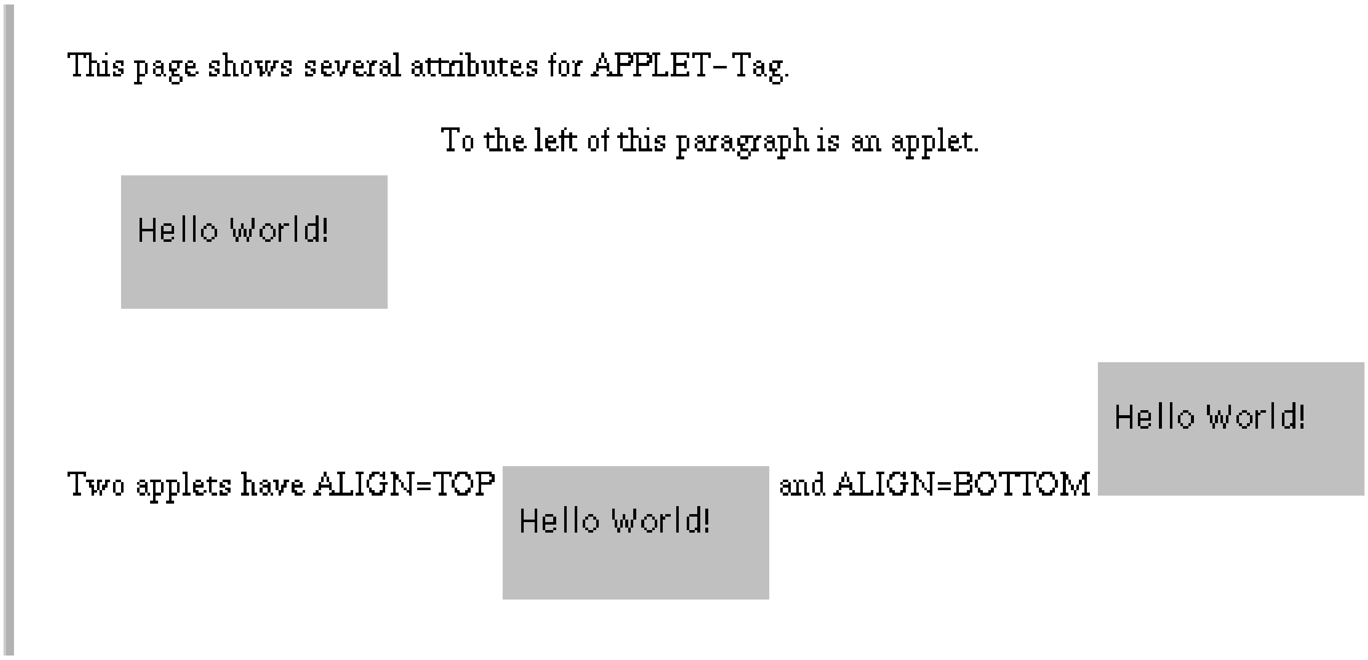
```
<APPLET CODE="MoreHelloApplet" WIDTH=300
HEIGHT=50>
<PARAM NAME=name VALUE="Qian">
<PARAM NAME=size VALUE="30">
Hello to whoever you are!
</APPLET>
```

- Die Applet-Klasse

```
import java.awt.*;
import java.applet.Applet
  // Fortsetzung auf der nächsten Folie
```

Weitere Attribute eines <APPLET>-Tags - Fortsetzung

• Der entsprechende Netscape Browser



• Anmerkungen

- Werte für `ALIGH`: `LEFT`, `RIGHT`, `TOP`, `MIDDLE`, `BOTTOM`, ...

- `HSPACE` und `VSPACE` beschreiben die leeren Raum *um* das Applet
  (`HSPACE` links und rechts, `VSPACE` oben und unten)

- Der `<BR>`-Tag, das `CLEAR` Attribut und die Werte `ALL`, `LEFT` und `RIGHT`.

Weitere Attribute eines <APPLET>-Tags

- Teil einer HTML-Datei

```
<P>
This page shows several attributes for AP-
PLET-Tag.
<P>
<APPLET CODE="HelloWorldApplet" WIDTH=300
HEIGHT=200
ALIGN=LEFT VSPACE=20 HSPACE=50></APPLET>
To the left of this paragraph is an applet.
<BR CLEAR=ALL>
The following applet has ALIGN=TOP
<APPLET CODE="HelloWorldApplet" WIDTH=300
HEIGHT=200
ALIGN=TOP></APPLET>
and the following has ALIGN=BOTTOM
<APPLET CODE="HelloWorldApplet" WIDTH=300
HEIGHT=200
ALIGN=BOTTOM></APPLET>
```

<APPLET>-Tag

• Teil einer HTML-Datei

```
<APPLET CODE="HelloAgainApplet.class"
 WIDTH=150 HEIGHT=25>
There would be an applet here if your
browser supported Java.
</APPLET>
```

• Anmerkungen

- `HelloAgainApplet.class` muß im gleichen Verzeichnis wie die HTML-Datei
  Sonst `CODEBASE=Verzeichnis` benutzen.

- `WIDTH` und `HEIGHT` - die Größe des Applets

- Text zwischen `<APPLET>` und `</APPLET>` für nicht-Java-fähige WWW-Browser

Das “HelloWoldApplet” Beispiel

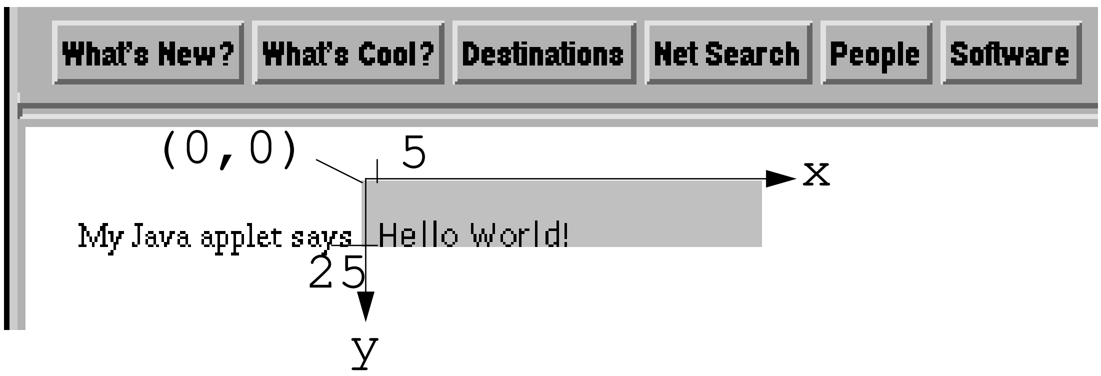
• Beispiel

```
import java.applet.Applet;
import java.awt.Graphics;

public class HelloWorldApplet
                              extends Applet{
  public void paint(Graphics g){
       g.drawString(“Hello World!”, 5, 25);
  }
}
```

- Überschreiben von `void paint(Graphics g)`

- Position beim Aufruf
 `g.drawString("Hello World!", 5, 25);`



Einige Methoden der Applet Klasse

- Fall 2: Vor dem Verlassen eines Applets
  (Z.B. wenn Browser auf eine andere Seite geht.)

  Aufruf der Methode `void stop()`
  (Stoppen des Applets)

- Fall 3: Wenn ein Applet wieder aktiviert werden soll
  (Z.B. nach Rückkehr zu einer früher unterladenen WWW-Seite)

  Aufruf der Methode `void start()`
  (Starten des Applets)

- Fall 4: Bevor ein Applet zerstört wird
  (Z.B. immer vor der Ausführung von Exit oder Reload)

  Aufruf der Methode `void destroy()`
  (Zerstören des Applets)

- Fall 5: Wenn etwas auf dem Schirm angezeigt werden soll
  (Z.B. wenn man Refresh des Browser-Fensters ausführt)

  Aufruf der Methode `void paint(Graphics g)`
  (Anzeige des Applets)

  Anmerkungen:
  (1) `import java.awt.Graphics` nicht vergessen!
  (2) Der aktuelle Parameter für `g` wird vom Browser erzeugt.

- Die vordefinierte Klasse `java.applet.Applet`

- Benutzer-definierte Applets als Objekte einer Unterklasse von `Applet`

- Die Unterklassen können die Methoden von Applet überschreiben

- WWW-Seite enthält Verweise auf solche Unterklassen.

- WWW-Browser ruft Konstruktoren und Methoden der Unterklasse auf

- Fall 1: Erstmals Unterladen eines Applets
(Z.B. erstmals Load oder Reload der WWW-Seite)

(1) Erzeugen eines Objektes für jeden Applet-Verweis

(2) Aufruf der Methode `void init()`
(Initialisierung des Applets)

(3) Aufruf der Methode `void start()`
(Starten des Applets)

Java API

- Java API (Application Programming Interface) besteht aus acht Paketen:

`java.applet`	Applets
`java.io`	Ein- und Ausgaben
`java.lang`	Grundfunktionen von Java
(`import java.lang` ist immer implizit!)	
`java.net`	Netzwerkfunktionen
`java.util`	Nützliche Datentypen
`java.awt`	Grafiken, Texte, Fenster und GUIs `awt` - Abstract Windowing Toolkit
`java.awt.image`	Bitmap-Images
`java.awt.peer`	GUI Toolkit

(Siehe `/usr/local/java/src/java`)

Kapitel 3

Programmieren mit dem Java API