

PostScript™-Graphik

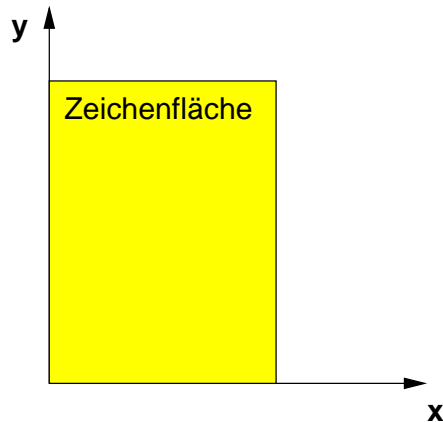
Jens Pönisch*

2003-02-14

*poenisch@isym.tu-chemnitz.de

Koordinatensystem

PostScript™ benutzt im (Gegensatz zu X11 und Windows) ein positiv orientiertes Koordinatensystem.



Standardeinheit ist *Punkt* (= 1/72" $\hat{=}$ 0.35 mm)

Umrechnung: `/mm {72 mul 25.4 div} def`

Verwendung: `10 mm 10 mm moveto`

Im weiteren verwenden wir Punkte.

Graphics state

Der *Graphics state* sammelt die aktuellen Einstellungen, die für das nachfolgende Rendern benutzt werden:

- Aktuelle Position (*current point*)
- Aktueller Pfad
- Farbe
- Linienstärke
- Linenstil, Linienenden
- Font
- ...

Er kann gespeichert und später wiederverwendet werden:

```
gsave grestore
```

Pfade

Gezeichnet wird, indem zunächst ein Pfad konstruiert wird. Dieser besteht aus einzelnen Segmenten, die nicht zusammenhängen müssen.

Nach Setzen der Zeichenparameter (Farbe, Dicke, ...) wird der Pfad verwendet:

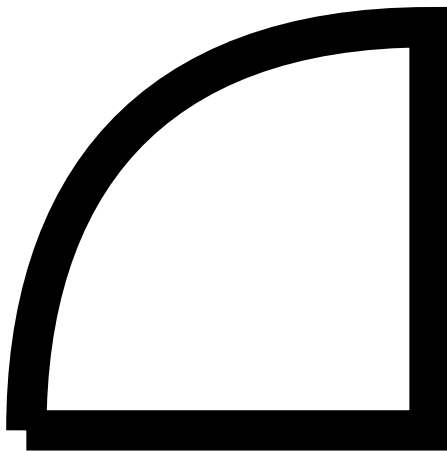
- Nachzeichnen: `stroke`
- Füllen (geschlossener Pfad): `fill`
- Ausschneiden (geschlossener Pfad): `clip`

Nach dem Rendern wird der Pfad gelöscht (`newpath`).

Zeichnen von Pfaden

- Position verändern: `x y moveto` oder `dx dy rmoveto`
- Line ziehen: `xneu yneu lineto` oder `dx dy rlineto`
- Beziérkurve: `x1 y1 x2 y2 xneu yneu curveto`
- Pfad schließen: `closepath`

Beispiel 1

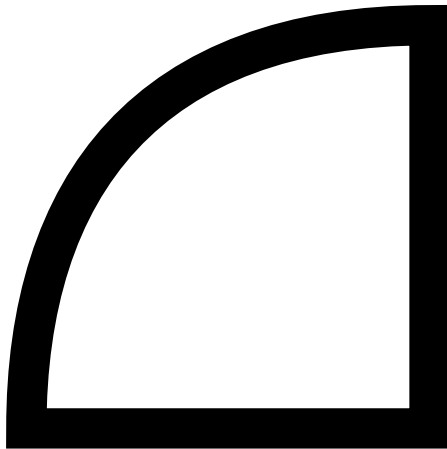


```
%!PS-Adobe-3.0
%%BoundingBox: 0 0 120 120

10 10 moveto % Anfangspunkt setzen
110 10 lineto % Linie ziehen
0 100 rlineto % relative Linie ziehen -> 110 110
40 110 10 70 10 10 curveto % Kurve ziehen
10 setlinewidth
stroke % Zeichnen
%showpage % entfällt, da encapsulated
```

Problem: „Ecke“ am Anfangspunkt

Beispiel 2



```
%!PS-Adobe-3.0
%%BoundingBox: 0 0 120 120

10 10 moveto % Anfangspunkt setzen
110 10 lineto % Linie ziehen
0 100 rlineto % relative Linie ziehen -> 110 110
40 110 10 70 10 10 curveto % Kurve ziehen
closepath % Pfad schließen
10 setlinewidth
stroke % Zeichnen
%showpage % entfällt, da encapsulated
```

closepath wandelt die Ecke auf den eingestellten Eckenstil (*linejoin*) um.

Beispiel 3



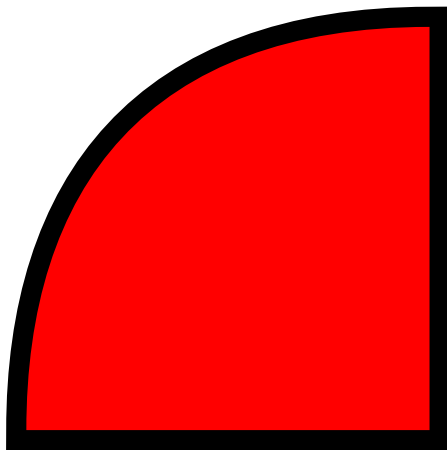
```
%!PS-Adobe-3.0
%%BoundingBox: 0 0 120 120

10 10 moveto % Anfangspunkt setzen
110 10 lineto % Linie ziehen
0 100 rlineto % relative Linie ziehen -> 110 110
40 110 10 70 10 10 curveto % Kurve ziehen
closepath % Pfad schließen
1.0 0.0 0.0 setrgbcolor % Farbe setzen
fill % Füllen
%showpage % entfällt, da encapsulated
```

Fläche wird gefüllt.

Beispiel 4

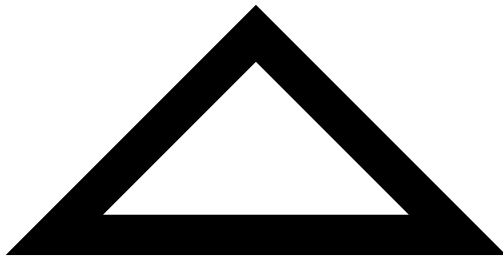
Sollen gleichzeitig Rand nachgezogen und die Fläche gefüllt werden, wird der Pfad zweimal benötigt. Mittels `gsave` kann er gespeichert (und damit dupliziert) werden.



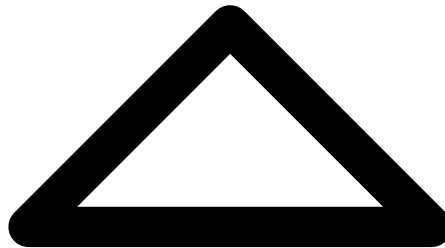
```
%!PS-Adobe-3.0
%%BoundingBox: 0 0 120 120

10 10 moveto 110 10 lineto 0 100 rlineto
  40 110 10 70 10 10 curveto
closepath % Pfad schließen
gsave      % Pfad speichern
10 setlinewidth stroke
grestore  % Pfad zurückholen
1.0 0.0 0.0 setrgbcolor % Farbe setzen
fill % Füllen
```

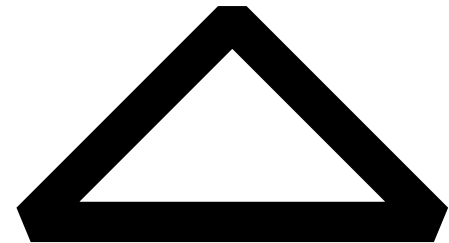
Linejoin



0 `setlinejoin`
Default (Miter)



1 `setlinejoin`
Abgerundete Ecken
(Round)

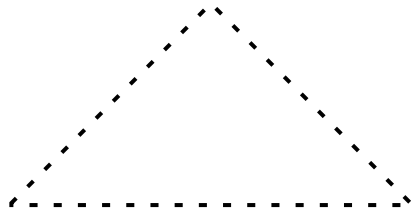


2 `setlinejoin`
Abgeschnittene
Ecken (Bevel)

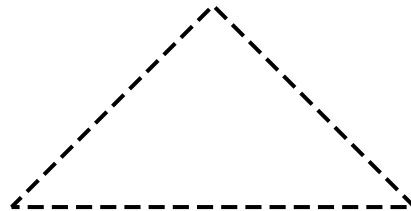
Analog für Enden einzelner Linien: `nr setlinecap`

Strichlinien

Angabe einer Matrix mit den Längen von Strich und Lücke sowie einer Verschiebung für den Anfang.



`[2 4] 1 setdash`



`[4 2] 2 setdash`

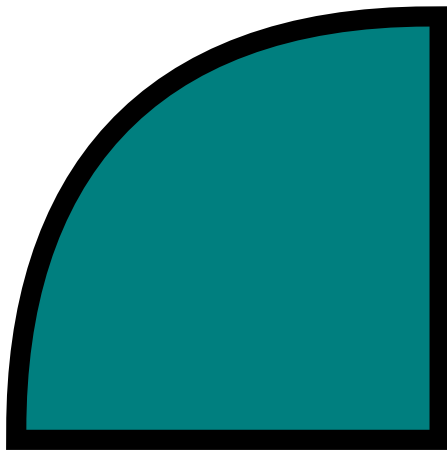


`[1 2 4 2] 0.5 setdash`

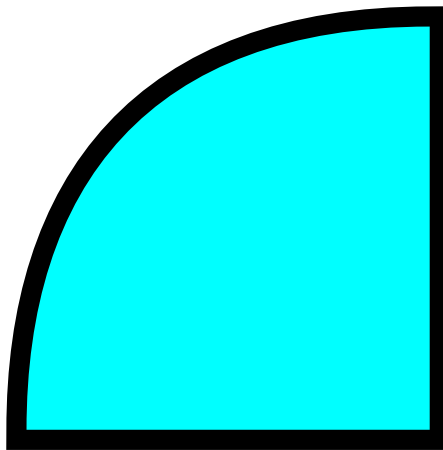
Farben

Farbmodelle: Graustufen, RGB, CMYK, ...

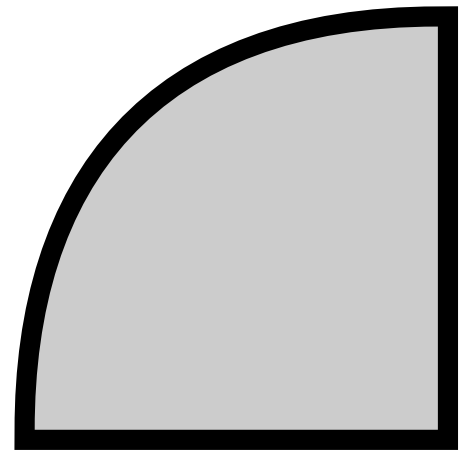
Farbangabe durch entsprechende Anzahl von Werten zwischen 0 und 1



```
0.0 0.5 0.5  
setrgbcolor
```



```
1.0 0.0 0.0 0.0  
setcmykcolor
```



```
0.8 setgray
```

Transformationen

Problem: PostScript™ kann nur Kreise, aber keine Ovale zeichnen.

`x y radius startwinkel endwinkel arc`

Lösung: Koordinatentransformationen

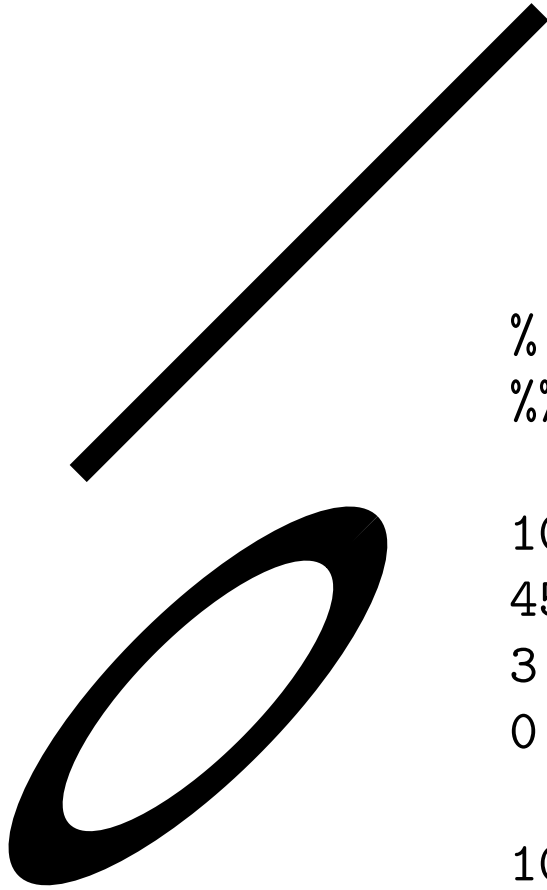
- Verschiebung des Ursprungs: `dx dy translate`
- Skalierung: `sx sy scale`
- Drehung: `winkel rotate`

Es wird eine Transformationsmatrix (CTM - current transformation matrix) gesetzt: `[a b c d dx dy]`:

$$x' = a x + b y + dx$$

$$y' = c x + d y + dy$$

Beispiel



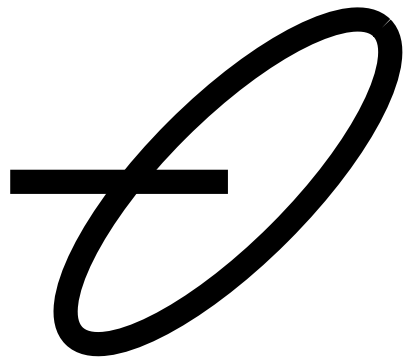
```
%!PS
%%BoundingBox: 0 0 200 200

100 100 translate
45 rotate
3 1 scale
0 0 30 0 360 arc    % Kreis mit Radius 30
                    % in x-Richtung skaliert
10 setlinewidth stroke
% Testlinie
10 100 moveto 100 100 lineto stroke
```

Problem(?): Ungleiche Linienstärke, Koordinatensystem für nächste Operation zurücksetzen.

CTM

Lösung der Probleme: Zurücksetzen der Transformation nach dem Festlegen des Pfades, aber vor dem Zeichnen.



```
%!PS
%%BoundingBox: 0 0 200 200

[1 0 0 1 0 0] currentmatrix
% legt CTM auf Stack, [1 0 0 1 0 0] ist Dummy
100 100 translate 45 rotate 3 1 scale
0 0 30 0 360 arc % Kreis mit Radius 30
% Hole alte CTM vom Stack und setze CTM
setmatrix
10 setlinewidth stroke
10 100 moveto 100 100 lineto stroke
```

Text

Textausgabe: (String) show

Zuvor: Position setzen (`moveto`) und Font auswählen:

- Font selbst: `/Helvetica findfont`,
- Größe: `24 scalefont`,
- im Graphics state setzen: `setfont`

Beispiel

Hallo

Problem: Umlaute

Hälo 250/

```
%!PS
%%BoundingBox: 0 0 100 50
```

```
10 10 moveto
/Helvetica findfont 24 scalefont setfont
(Hallo) show
```

```
%!PS
%%BoundingBox: 0 0 100 50
```

```
10 10 moveto
/Helvetica findfont 18 scalefont setfont
(Hälo 250/ ) show
```

Encoding

Adobe™ benutzt eigenes Font-Encoding mit max. 256 Zeichen, was jedoch geändert werden kann.

Umlaute sind nicht zulässig, sondern müssen oktal geschrieben werden.

```
%!PS
%%BoundingBox: 0 0 100 50
```

```
Hællø 250æ 10 10 moveto
/Helvetica findfont 18 scalefont setfont
(H\36111ø 250\250) show
```

ä ist nicht im Adobe-Standard-Encoding enthalten!

ISO-Encoding 1

```
/ISOLatin1Encoding [  
  ...  
  /space /exclam /quotedbl ...  
  /at /A /B /C /D /E /F /G ...  
  /dotlessi /grave /acute /circumflex /tilde /macron /breve /dotacc  
] def
```

Hällo 250ä

```
/ISOEncode {  
  dup length dict begin  
    {1 index /FID ne {def} {pop pop} ifelse} forall  
    /Encoding ISOLatin1Encoding def  
    currentdict  
  end  
  /Temporary exch definefont  
} bind def
```

```
10 10 moveto  
/Helvetica findfont 18 scalefont ISOEncode setfont  
(H\344llo 250\244) show
```

ISO-Encoding 2

Anzeige des €-Symbols: Ändern des Encoding-Vectors
Ersetze /currency durch /Euro – Font muss Symbol enthalten!

Hällo 250€

```
/ISOLatin15Encoding [ ...  
  /space /exclamdown /cent /sterling /Euro /yen /brokenbar /section  
  ... ] def  
/ISOEncode {  
  dup length dict begin  
    {1 index /FID ne {def} {pop pop} ifelse} forall  
    /Encoding ISOLatin15Encoding def currentdict  
  end  
  /Temporary exch definefont  
} bind def  
  
10 10 moveto  
/Helvetica findfont 18 scalefont ISOEncode setfont  
(H\344llo 250\244) show
```

Character Paths 1

Umwandlung einer Schrift in einen Pfad mit `false charpath`
Anwenden von Linienstärke, Füllung, ...



```
%!PS
%%BoundingBox: 0 0 100 50

10 10 moveto
/Helvetica findfont 44 scalefont setfont
(Hallo) false charpath
gsave
1 0.5 0.5 setrgbcolor fill
grestore
1 setlinewidth
[4 2] 0 setdash stroke
```

Character Paths 2

Nachzeichnen der Umriss mit verschiedenen Linienstärken und -farben

The word "Hallo" is displayed in a stylized, italicized font. Each letter is composed of two parallel outlines, one slightly offset from the other, creating a 3D or double-line effect. The font is slanted to the right.

```
%!PS
%%BoundingBox: 0 0 100 50

10 10 moveto
/HelveticaItalic findfont 44 scalefont setfont
(Hallo) false charpath
gsave
4 setlinewidth 0 setgray stroke
grestore
2 setlinewidth 1 setgray stroke
```

Character Paths 3

Nutzen der Buchstaben zum Ausschneiden einer Clipping Region.
Es wird eine schräge, sehr breite Linie mit der Buchstabenkontur ausgestanzt.



```
%!PS-Adobe-3.0
%%BoundingBox: 0 0 100 50
10 10 moveto
/HelveticaItalic findfont 44 scalefont setfont
(Hallo) false charpath
clip      % Schnittmaske
newpath % Neuen Pfad beginnen
0 0 moveto 100 50 lineto 150 setlinewidth
[2 2] 0 setdash 0 0 1 setrgbcolor stroke
```

Messen mit Character Paths

Textfeld farbig hinterlegen \Rightarrow Textgröße wird benötigt: pathbbox



```
%!PS-Adobe-3.0
%%BoundingBox: 0 0 150 50
/HelveticaItalic findfont 44 scalefont setfont
(Hallo) dup
10 10 moveto
gsave
false charpath flattenpath pathbbox
3 index 3 index moveto 1 index 3 index lineto
2 copy          lineto 3 index 1 index lineto
pop pop pop pop closepath
1 0 0 setrgbcolor fill
grestore
1 1 1 setrgbcolor show
```

Verbesserung: Box vergrößern



```
%!PS-Adobe-3.0
%%BoundingBox: 0 0 150 50
/HelveticaItalic findfont 44 scalefont setfont
(Hallo) dup
10 10 moveto
gsave
false charpath flattenpath pathbbox
3 index 5 sub 3 index 5 sub moveto
1 index 5 add 3 index 5 sub lineto
1 index 5 add 1 index 5 add lineto
3 index 5 sub 1 index 5 add lineto
pop pop pop pop closepath
1 0 0 setrgbcolor fill
grestore
1 1 1 setrgbcolor show
```

Stringbreite

Exakt: pathbbox, vereinfacht: stringwidth → x y

Font vorher setzen, y gibt den Versatz der Höhe in der Schrift an (= 0)!

Hallo
Ancient Europe!

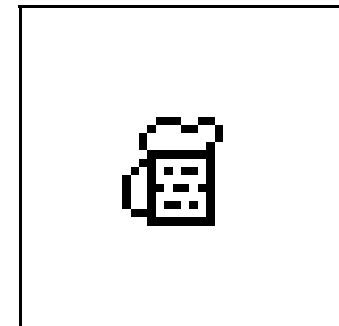
```
/HelveticaBoldItalic findfont
 24 scalefont setfont
100 45 (Hallo) center_show
100 15 (Ancient Europe!) center_show
100 0 moveto 100 70 lineto stroke
```

```
/center_show { % x y s -> --
  dup stringwidth pop
  2 div
  4 -1 roll exch sub
  3 -1 roll moveto
  show
} def
```

Einbinden von Bildern (Bitmaps)

Bild mit `bitmap` invers und an Senkrechte gespiegelt speichern.
Bytes paarweise vertauschen.

```
                                %!PS
#define bier_width 16           %%BoundingBox: 0 0 100 100
#define bier_height 16         /Bild
static unsigned char bier_bits[] < ffff ffff f8cf f737
= {                             eff7 efef f00f e7ef
    0xff, 0xff, 0xff, 0xff,     d52f b7ef b24f b7ef
    0xcf, 0xf8, 0x37, 0xf7,     b4af c7ef f00f ffff > def
    0xf7, 0xef, 0xef, 0xef,
    0x0f, 0xf0, 0xef, 0xe7,     50 50 translate
    0x2f, 0xd5, 0xef, 0xb7,     50 50 scale
    0x4f, 0xb2, 0xef, 0xb7,     16 16 1
    0xaf, 0xb4, 0xef, 0xc7,     [16 0 0 -16 8 8] % An x-Achse spiegeln
    0x0f, 0xf0, 0xff, 0xff};    { Bild }
                                image
```



Bilddaten

Die Bilddaten können verschieden eingebunden werden, hier als Hexstring im Dictionary gespeichert:

`<Hexadezimalstring>`

Der `image`-Befehl benötigt zunächst die Größe und Farbtiefe des Bildes. Die Transformationsmatrix beschreibt die notwendigen Parameter, um das Pixelbild auf ein 1x1-Quadrat zu transformieren.

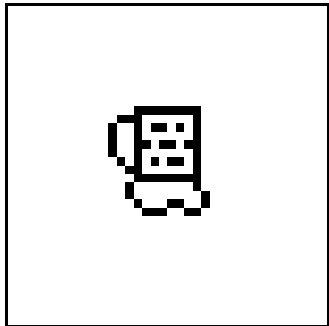
Üblich:

`[breite 0 0 höhe 0 0]` (rechtsdrehendes Koordinatensystem)

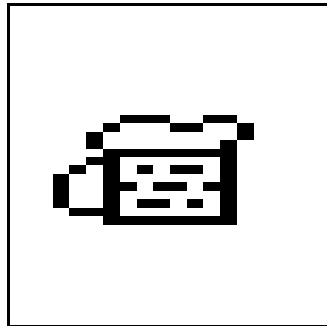
`[breite 0 0 -höhe 0 höhe]` (linksdrehendes Koordinatensystem)

Beispiele

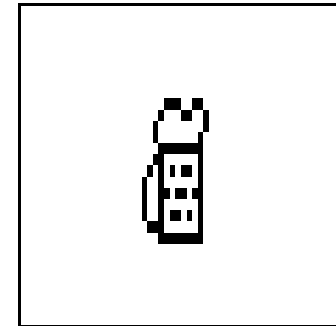
Verschiedene Transformationsmatrizen



[16 0 0 16 8 8]



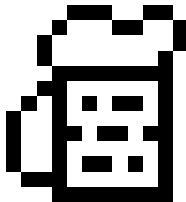
[8 0 0 16 8 8]



[24 0 0 -12 8 8]

Daten direkt lesen

Problem: Stack-Überlauf bei großen Bildern.
Alternative: String aus File direkt lesen.



```
%!PS-Adobe-3.0
%%BoundingBox: 0 0 100 100
50 50 translate 50 50 scale
16 16 1
[16 0 0 -16 8 8] % An x-Achse spiegeln
{ currentfile 8 string readhexstring pop }
image
ffff ffff f8cf f737
eff7 efef f00f e7ef
d52f b7ef b24f b7ef
b4af c7ef f00f ffff
% Bildaten 8 Byte/Zeile
```

– ENDE PostScript™-Graphik –