Befehle Turtle

Die Turtlebefehle für Scripte werden von C++ Grafiken des Typs 'Script xyz' zur Verfügung gestellt.

Hilfedateien

Scripte:	2_Scripte.pdf
Befehle Grafiken:	4_Grafik.pdf
Farben:	5_Farben.pdf

Funktionen mit Parametern: Parameter ohne Defaultwerte müssen immer angegeben werden.
Optionale Parameter können weggelassen werden.

Beispiel

Befehl: `t.setPen(Farbe="red" , Stärke=-1 , Stil=-1);`

Mögliche Aufrufe:

```
t.setPen();                    // Farbe="red", Stärke=-1, Stil=-1
t.setPen( "green" );           // Farbe="green", Stärke=-1, Stil=-1
t.setPen( "green" , 1.2 );     // Farbe="green", Stärke= 1.2, Stil=-1
t.setPen( "green" , 1.2 , 2 ); // Farbe="green", Stärke= 1.2, Stil=2
```

Turtle Initialisierungen

Verwendung nur in init(). Alle setPage() Befehle setzen die Hintergrundfarbe Brush und die Stiftfarbe Pen.

`t.setPage();`	Papierformat und Rand aus Einstellungen.
`t.setPage(din=255);`	Querformat: dinA 0 - 9, Hochformat: + 10 255 aus Einstellungen.
`t.setPage(x0, y0, din=255);`	(x0,y0) sind die Koordinaten der linken oberen Ecke in mm.
`t.setMargin(RandX=-1,RandY=-1);`	Papierränder setzen. -1 aus Einstellungen. -2 Zeichenfläche quadratisch.

`t.setAutoPaint( on=true);`	Automatisches Neuzeichnen. Für langwierige Grafiken.
`t.setAntiAliasing(on=true);`	Linen sauberer zeichnen. Ausfürung langsamer!

Turtle Pen und Brush

`t.setNoStyle();`	Pen- und Brushstyle =0. Turtle zeichnet nicht. Aktuellen Pen und Brush merken.

`t.setPen();`	Letzten Pen wieder verwenden.
`t.setPen( Farbe="", Stärke=-1, Stil=-1 );`	Farbe siehe Hilfe Farben, Stärke in mm (Minimum -1), Stil 0-6 (siehe Grafiktools).
`t.setPenWidth( Stärke=-1, CapStyle=-1 );`	CapStyle 1=FlatCap, 2=SquareCap oder 3=RoundCap.

`t.setBrush();`	Letzten Brush wieder verwenden
`t.setBrusch( Farbe="", Stil=-1, Transparenz=-1 );`	Farbe siehe Hilfe Farben, Füllstil 0-24 (siehe Grafiktools), Transparenz 0-255.
`t.setGradient(x1, y1, x2, y2, Farbe1, Farbe2 );`	Linearen Farbverlauf von Farbe1 in P1(x1,y1) nach Farbe2 in P2(x2,y2) festlegen.
`t.setGradient(t, Farbe);`	0<t<1: Farbe im Zwischenpunkt P1 + t*(P2-P1) festlegen. Verwendung mit t.setBrusch("", 15); !

Turtle bewegen

`t.turn( -7.5 );`	-7.5° im Uhrzeigersinn drehen.
`t.turnTo( w );`	Turtle-Richtung w in Grad einstellen. Richtung der x-Achse ist 0°.
`t.turnTo( x, y);`	Turtle-Richtung w von Punkt lastX nach P(x,y) einstellen.
`t.turnTo("P1");`	Turtle-Richtung w von Punkt lastX auf Punkt oder Polygon "P1" einstellen.

`t.goTo( x, y );`	Turtle ohne Pen auf Punkt (x,y) bewegen.
`t.move( 10.1 );`	Turtle um 10.1 mm bewegen und mit Pen zeichnen.
`t.moveTo( 10, 20 );`	Turtle auf Punkt (10,20) bewegen und mit Pen zeichnen.
`t.moveTo("P1");`	Turtle auf Punkt oder Polygon "P1" mit Pen bewegen.
`goTo("P1", Index=0);`	Turtle auf Punkt oder Polygon "P1" ohne Pen bewegen. Bei einem Polygon bestimmt der Index den Punkt. Index ist 0,1, bis n oder -1,-2, bis -(n+1). 0 ist der erste Punkt, -1 der letzter Punkt. Siehe 'Befehle Grafik', 'Eigene Befehle'.

t.drawPoint(x, y); Punktmarke für Punkt (x,y) zeichnen.

Turtle Rechtecke

t.rectangle(dx, dy); Rechteck an Turleposition mit Turlerichtung ausgeben.

t.rahmen(); Rahmen um die Zeichenfläche nochmals zeichnen. Pen vorher wählen.

Turtle Polygonmodus

Im Polygonmodus werden die jeweiligen Endpunkte der Bewegungen gespeichert im Polygon gespeichert.

t.beginPolygon(); Polygon löschen und Polygonmodus einschalten. Alle folgenden Bewegungen werden gespeichert.

t.endPolygon(FillMode=0); Polygonmodus beenden.

FillMode: 0 = geschlossen, OddEvenFill
1 = geschlossen, WindingFill
2 = geschlossen, nicht gefüllt
3 = offen, nicht gefüllt

t.movePolygon(x, y); Alle Polygonpunkte durch die verschobenen Punkte ersetzen. Schiebvektor: Ursprung auf P (x,y).

t.drawPolygon(); Polygon mit aktuellem Stift und Brush an der aktuellen Turtleposition zeichnen. Vor dem Zeichnen in Turtlerichtung drehen. Polygon bleibt unverändert.

t.clrPolygon(); Polygon löschen.

t.rectPolygon(dx, dy, FillMode=0); Rechteckiges Polygon an Turleposition in Turtlerichtung erzeugen.

t.trianglePolygon(c, wb, b, FillMode=0); Dreieck ABC mit wb in Grad. A(lastX(),lastY()), B in Richtung lastW().

t.isoPolygon(r, n=-1, FillMode=0); n> 0: Regelmäßiges n-Eck, Radius r, Mittelpunkt (lastX(),lastY()), Richtung lastW().
n= -1: Kreisnäherung. -2 zweifache Punkteanzahl.

t.splinePolygon(dt=0.1); Polygon durch ein quadratisches Bezier Spline ersetzen.
epsilon<dt<1: Punktedichte pro Spline-Segment.

t.cPolygon(); Polygon durch ein C-Polygon ersetzen. Das C-Polygon besteht aus dem gegebenen Polygon und allen am Endpunkt gespiegelten Punkten.

Turtle Texte ausgeben

t.text("Text"); "Text" an der Turleposition mit -richtung ausgeben.
t.text(1, 2, "Text"); "Text" bei Punkt (1/2) mit Turtlerichtung ausgeben.
t.text(x0, y0, x1, y1, "Text"); "Text" im Rechteck(x0,y0,x1,y1) formatiert in Turlerichtung ausgeben.

t.setFont("FontName", size_mm, FixPitch=true, Weight=50); Schrifttyp setzen. Fontnamen siehe Einstellungen. Weight 0-99.

t.setWeight(Weight); Weight 0-99.

Turtle Bild anzeigen

t.bild("DateiName", w_mm=-1, h_mm=-1);; Bildatei "DateiName" an der Turleposition mit -richtung ausgeben.
w_mm=-1 : Breite in mm, -1 proportional
h_mm=-1 : Höhe in mm, -1 proportional

Turtle Status

t.lastX(); Aktuelle x-Koordinate der Turtle.
t.lastY(); Aktuelle y-Koordinate der Turtle.
t.lastW(); Aktuelle Bickrichtung der Turtle in Grad.

t.getL(); Betrag des Vektors (0,0)->(lastX(), lastY()).
t.getW(); Richtung des Vektors (0,0)->(lastX(), lastY()).

t.msgTurtle(); Aktuelle Einstellungen der Turtle in der Messagebox anzeigen.