

Mit dem Python-Modul `turtle` kann man ganz einfach anfangen, Bilder zu malen!

Um anzufangen, brauchen wir zuerst diesen Code:

```
1 from turtle import *
2 colormode(255)
3 # Hier kommt unser Code rein!
4 done()
```

Zwischen die Wahl des Farbmodus in Zeile 2 und der Anzeige des Bilds in Zeile 4 können wir jetzt unseren Code schreiben, um Bilder zu malen. Dafür gibt es folgende Funktionen:

Befehl	Funktion
<i>Bewegung</i>	
<code>forward(distance)</code>	Bewegt den Stift um <code>distance</code> nach vorne.
<code>backward(distance)</code>	Bewegt den Stift um <code>distance</code> rückwärts.
<code>right(angle)</code>	Dreht die Blickrichtung um <code>angle</code> nach rechts
<code>left(angle)</code>	Dreht die Blickrichtung um <code>angle</code> nach links
<code>goto(x, y)</code>	Bewegt den Stift an die Koordinaten <code>(x, y)</code> .
<i>Stiftkontrolle</i>	
<code>pendown()</code>	Setzt den Stift auf das Papier.
<code>penup()</code>	Hebt den Stift vom Papier.
<code>pensize(size)</code>	Setzt die Stiftgröße auf <code>size</code> .
<code>pencolor(color)</code>	Ändert die Stiftfarbe auf <code>color</code> . <code>color</code> kann ein String sein (wie z.B. "red") oder ein RGB-Tupel (wie (255, 20, 120)).
<i>Formen füllen</i>	
<code>begin_fill()</code>	Startet die Aufzeichnung einer Form
<code>end_fill()</code>	Beendet die Aufzeichnung der Form und füllt diese aus.
<code>fillcolor(color)</code>	Setzt die Füllfarbe auf <code>color</code> . <code>color</code> kann ein String sein (wie z.B. "red") oder ein RGB-Tupel (wie (255, 20, 120)).

Befehl	Funktion
<i>Hintergrund</i>	
<code>screensize(width, height)</code>	Legt die Größe des Bildes fest.
<code>bgcolor(color)</code>	Setzt die Hintergrundfarbe auf <code>color</code> . <code>color</code> kann ein String sein (wie z.B. "red") oder ein RGB-Tupel (wie (255, 20, 120)).
<code>bgpic(path)</code>	Setzt ein Hintergrundbild.
<i>Bild exportieren</i>	
<code>hideturtle()</code>	Blendet den Cursor aus (praktisch für den Bildexport).
<code>showturtle()</code>	Zeigt den Cursor wieder an.
<code>save(filename)</code>	Speichert das Bild unter dem Dateinamen <code>filename</code> . Achtung: Leider nur als PostScript-Datei (.ps) ¹

Alles klar soweit? Na dann: Viel Spaß beim Malen!

¹Falls `save()` nicht funktioniert, kann man auch `getcanvas().postscript(file=filename)` verwenden – das klappt häufiger! Die PostScript-Datei kann man mit folgendem Befehl unter Linux in ein PNG-Bild konvertieren: `convert -density 300 filename.ps filename.png`