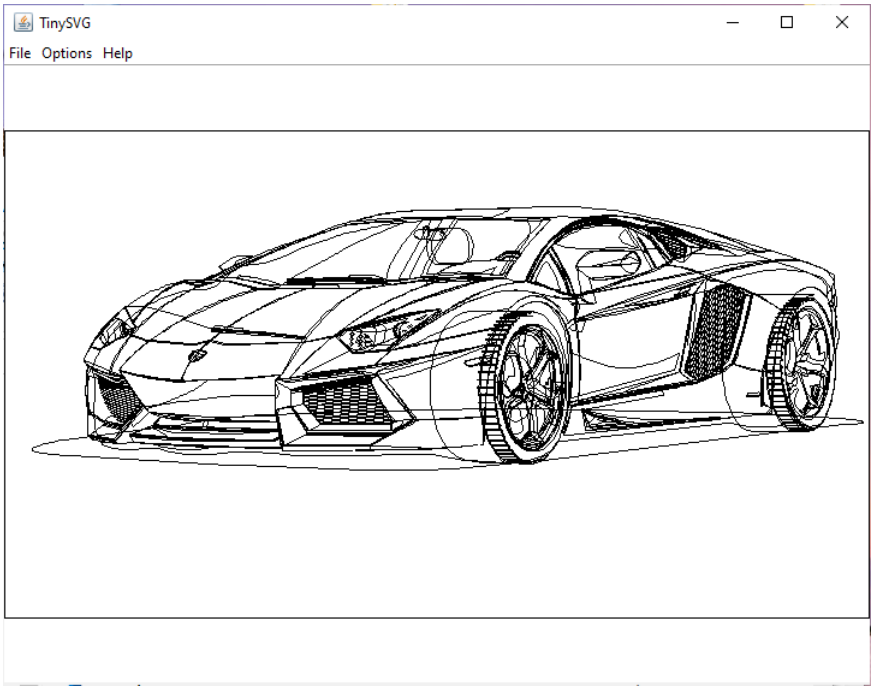
TinySVG

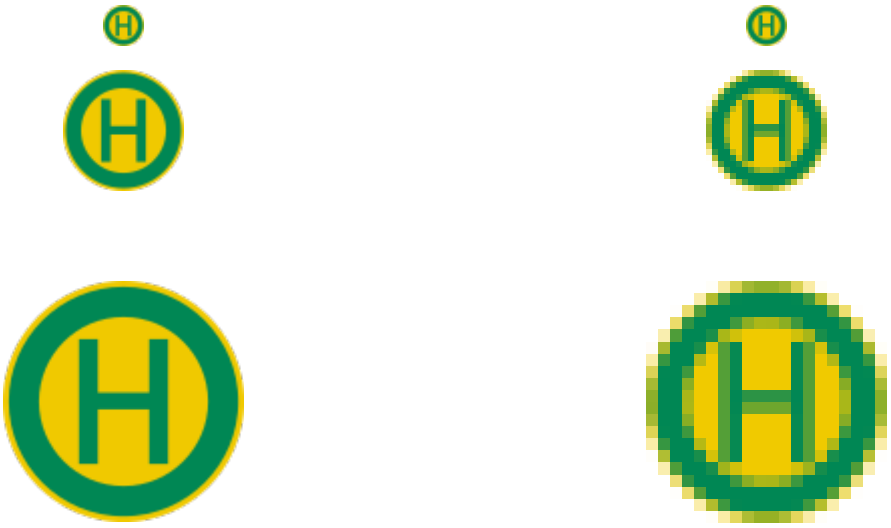


Stand: 16.10.2022

Inhalt

TinySVG

In der Computergrafik unterscheidet man zwei Arten von Bildern: Bitmap- und Vektorgrafiken. Erstere speichern die Farbe jedes einzelnen Bildpunktes in einem rechteckigen Raster, letztere speichern eine Beschreibung der Bildobjekte wie Linien, Kreise, Quadrate, usw. Möchte man die Bilder vergrößern oder verkleinern (skalieren), dann geht das nur mit Vektorgrafiken verlustfrei, denn die Bildobjekte können jeweils neu in die gewünschte Größe umgerechnet werden. Bei Bitmap-Bildern kennt man die „Treppenstufen“ bei schrägen Linien, die besonders bei Vergrößerungen sichtbar werden – je nach Vergrößerungsalgorithmus mehr oder weniger deutlich.



Vektorgrafik Bitmap-Grafik

Das Internet ist voll von Bildern; die meisten davon sind Bitmap-Grafiken. Es gibt aber auch viele Vektorgrafiken, und das Standardformat für Vektorgrafiken im Internet heißt SVG (**S**calable **V**ector **G**raphics = Skalierbare Vektorgrafik). Es handelt sich hierbei im Wesentlichen um ein XML-basiertes Format, in dem man Tags wie <line>, <circle>, <rect> oder <path> für die geometrischen Grundformen findet. Die Tags werden durch Attribute genauer beschrieben; diese geben z.B. Start- und Endpunkte, Radius, Breite, Höhe, Linienstärke etc. an. Das komplizierteste Attribut ist wohl das „d“-Attribut in <path>: Es beinhaltet einfache Grafikbefehle wie MoveTo oder LineTo, aber auch komplexere wie elliptische Bögen oder Bézier-Kurven. Damit lässt sich eine Vielzahl von Formen zeichnen.

Beispiel einer SVG-Datei:

```
<svg width="300" height="200" viewbox="0 0 300 200">
  <desc>Flagge Dänemarks</desc>
  <rect x="0" y="0" width="300" height="200" fill="#E31836" />
    <path d="m80,0 h40 v80 h180 v40h-180 v80 h-40 v-80 h-80 v-40 h80z" fill="white" />
</svg>
```

SVG-Grafiken zum Herunterladen findet man u.a. hier:

Frei	**Kommerziell**
https://freesvg.org	https://www.shutterstock.com
https://publicdomainvectors.org	
https://freesvgclipart.com/	
https://www.vecteezy.com/	
https://www.freevector.com/	
https://www.123freevectors.com	
https://www.vector4free.com/	

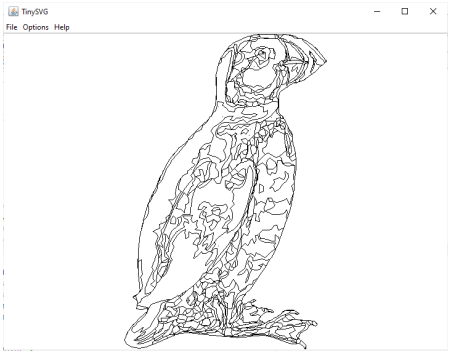
Man kann SVG-Vektorgrafiken auch selbst erstellen: das Open-Source-Programm **Inkscape** ist empfehlenswert. Inkscape kann auch Bitmap-Bilder in Vektorgrafiken umwandeln (was aber nicht immer perfekt gelingt und einiges Ausprobieren erfordert).

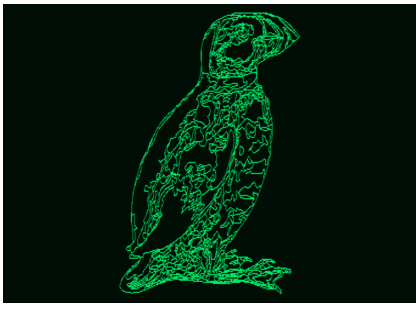
SVG ist ein komplexes Format, das hinsichtlich der Darstellungsmöglichkeiten keine Wünsche offen lässt. Die Komplexität des Formats wirft die Frage auf, wie man SVG-Grafiken auf einem 8-Bit-Computer der 1980er Jahre mit eingeschränkten Grafikmöglichkeiten und knappem Hauptspeicher zur Anzeige bringen kann. Das direkte Einlesen und Interpretieren von SVG-Dateien ist schon allein wegen des Speicherbedarfs kaum möglich.

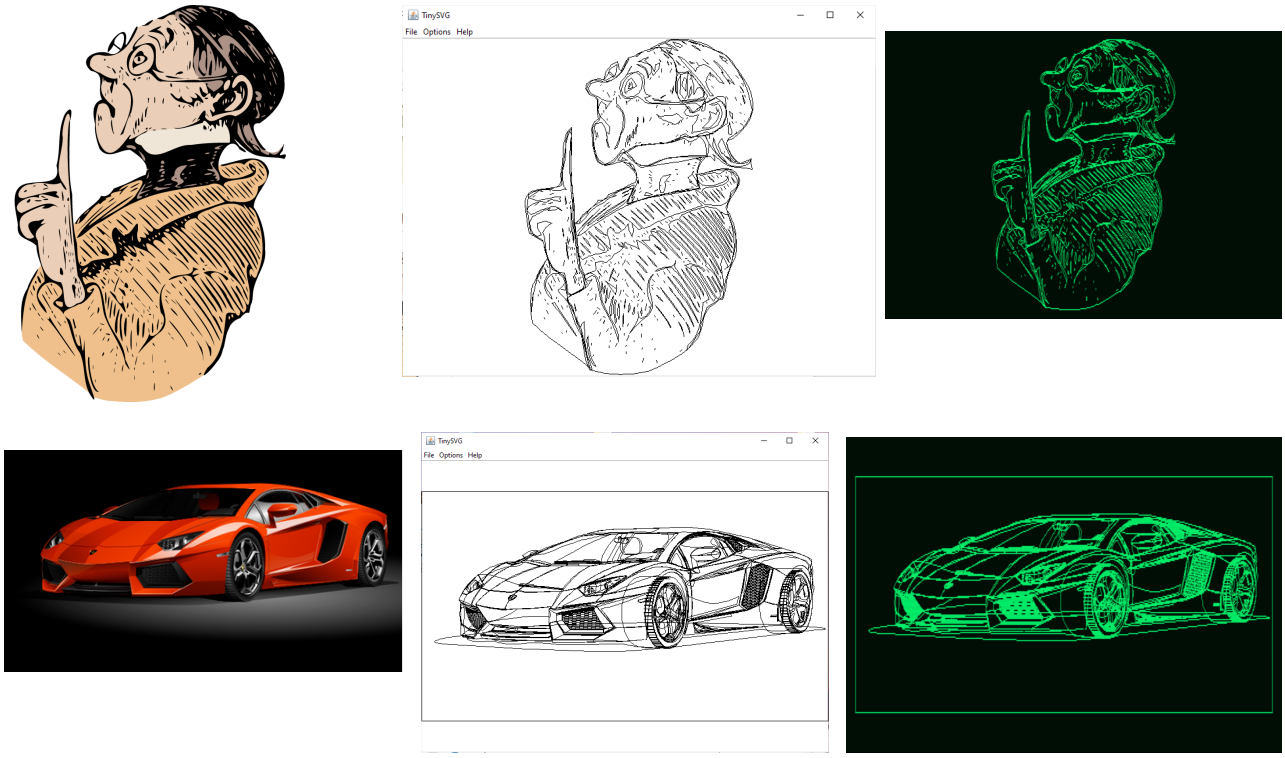
Der Ansatz ist daher zweistufig: zuerst wird die SVG-Grafik auf dem PC in ein einfaches Vektorformat umgewandelt, das nur Grafikbefehle enthält, die auf einem 8-Bit-Computer interpretiert werden können. Dabei werden alle SVG-Spezialitäten weggelassen, die auf einem monochromen Bildschirm nicht dargestellt werden können. Dann liest ein Programm auf dem 8-Bit-Computer das vereinfachte Vektorformat ein und zeichnet die Grafik.

Für den ersten Schritt habe ich ein Java-Programm namens **TinySVG** entwickelt, das mittlerweile mehrere 8-Bit-Plattformen unterstützt. Aktuell, d.h. in Version 1.0, sind noch nicht alle SVG-Features implementiert, aber die meisten SVG-Dateien lassen sich bereits gut umwandeln. Den zweiten Schritt übernimmt ein Programm auf der jeweiligen Zielplattform, oder die von TinySVG ausgegebene Datei enthält selbst den Code zur Darstellung der Grafik.

Die folgenden Beispiele zeigen SVG-Grafiken, wie sie im Internet Browser, in TinySVG und via vecread auf der Schneider Joyce dargestellt werden:

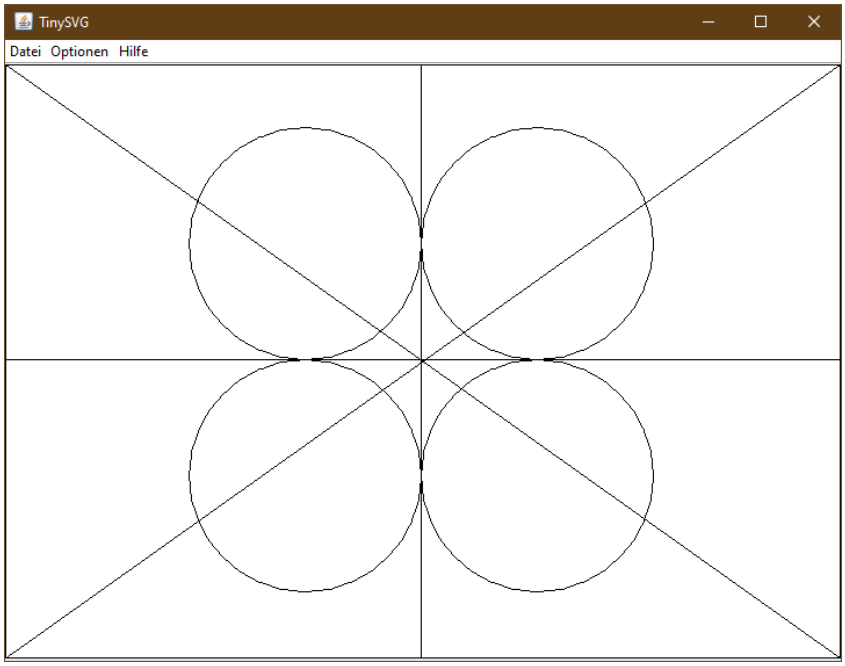






Bedienung

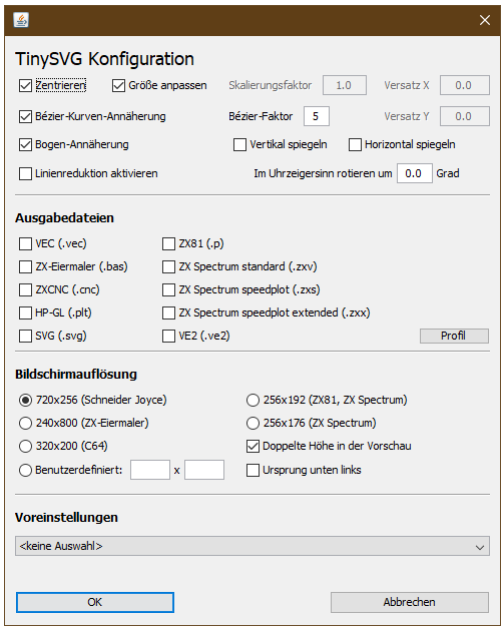
Zur Ausführung von TinySVG muss eine Java-Runtime ab Version 8 auf dem PC installiert sein. Das Programm besteht aus einer JAR-Datei, die man einfach per Doppelklick starten kann.



Zu Beginn sollte man die gewünschte Sprache einstellen, falls das Menü nicht bereits in der korrekten Sprache erscheint: **Optionen > Deutsch** oder **Optionen > Englisch** bzw. **Options > German** oder **Options > English**. Im folgenden werden die deutschen Menüs verwendet.

Konfiguration

Anschließend stellt man die Konfiguration via **Optionen** > **Konfiguration** ein.



Die Option **Zentrieren** sollte aktiviert sein; dadurch wird die Grafik in der Mitte des Vorschaufensters angezeigt.

Auch **Größe anpassen** sollte aktiviert sein, damit die Grafik auf die volle verfügbare Höhe und Breite angepasst wird. Die Anpassung geschieht unter Beibehaltung des Seitenverhältnisses, d.h. die Grafik wird nicht gestreckt oder gestaucht.

Bézier-Kurven-Annäherung rechnet in <path>-Tags enthaltene Bézier-Kurven in Liniensegmente um. Die Option sollte aktiviert sein, sonst werden Bézier-Kurven durch gerade Linien ersetzt, was durchaus zu interessanten Effekten führen kann, im Allgemeinen aber unerwünscht ist. Der **Bézier-Faktor** verändert die Anzahl der erzeugten Liniensegmente. Je kleiner der Wert ist, desto mehr Linien werden erzeugt und desto glatter wird die Kurve. Die Datei wird dann aber größer und das Zeichnen der Grafik dauert länger. Der Standardwert von 10 passt meist ganz gut.

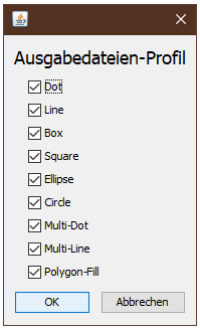
Die **Bogen-Annäherung** funktioniert ganz ähnlich, allerdings für Kreis- oder Ellipsenbögen. Auch diese Option sollte aktiviert sein.

TinySVG kann die Anzahl der Linien reduzieren, indem es unnötige Linien entfernt. Das sind zum einen verbundene Linien mit nur einem Pixel Länge. Wenn zwei Linien verbunden sind und denselben Steigungswinkel haben, dann werden sie zu einer Linie zusammengefasst. Die Option **Linienreduktion aktivieren** sollte i.d.R. eingeschaltet bleiben.

Bei Bedarf kann die Grafik gespiegelt werden; dazu kann man die Optionen **Vertikal spiegeln** oder **Horizontal spiegeln** aktivieren. Auch eine Drehung der Grafik ist möglich. Der Drehwinkel wird in Grad angegeben, ggf. mit Dezimalpunkt (nicht -komma). Die Einstellung **Im Uhrzeigersinn rotieren um** ist dafür zuständig.

Ausgabedateien

Unter **Ausgabedateien** kann man wählen, welche Vektordateien TinySVG erzeugen soll. Für die Schneider Joyce sollte **VEC (.vec)** oder besser noch **VE2 (.ve2)** aktiviert sein. Das sind Formate, die das Programm **vecread** in der aktuellen Version lesen kann. Man kann aber z.B. auch HP-GL-Dateien mit **HP-GL (.plt)** erzeugen, die dann z.B. mit dem **HP-GL Interpreter** angezeigt werden können. Für andere Zielplattformen wählt man andere Optionen (s. Seite 8).



TinySVG kann auch die aus der SVG-Datei extrahierte Grafik wiederum als SVG-Datei speichern. Dazu wählt man die Option **SVG (.svg)**.

Über die Schaltfläche **Profil** kann man festlegen, welche Grafikobjekte TinySVG in den Ausgabedateien erzeugen soll. Diese Grafikobjekte muss das Anzeigeprogramm auf der Zielplattform verstehen. Normalerweise muss die Standardeinstellung nicht geändert werden.

Bildschirmauflösung

Über die **Bildschirmauflösung** wird die Auflösung der ausgegebenen Vektoren bestimmt. Es gibt einige Optionen für bestimmte Zielplattformen, die man direkt wählen kann. Alternativ gibt man die X- und Y-Auflösung mittels **Benutzerdefiniert** selbst ein.

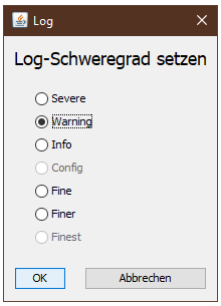
Eine Besonderheit ist die Option **Doppelte Höhe in der Vorschau**. Sie ist besonders für die Schneider Joyce interessant: da das Pixel-Seitenverhältnis (PAR = pixel aspect ratio) nahezu 1:2 beträgt, bekommt man so ein unverzerrtes Bild angezeigt. Die tatsächliche Auflösung der Joyce von 720x256 wird also als 720x512 angezeigt.

Voreinstellungen

Alle Einstellungen unter **Ausgabedateien** und **Bildschirmauflösung** kann man auch ganz einfach über die **Voreinstellungen** auswählen. Man wählt einfach die gewünschte Zielplattform und die empfohlenen Optionen werden gesetzt. Die Hauptoptionen ganz oben werden hierbei nicht verändert - nur die Optionen unter **Ausgabedateien** und **Bildschirmauflösung**.

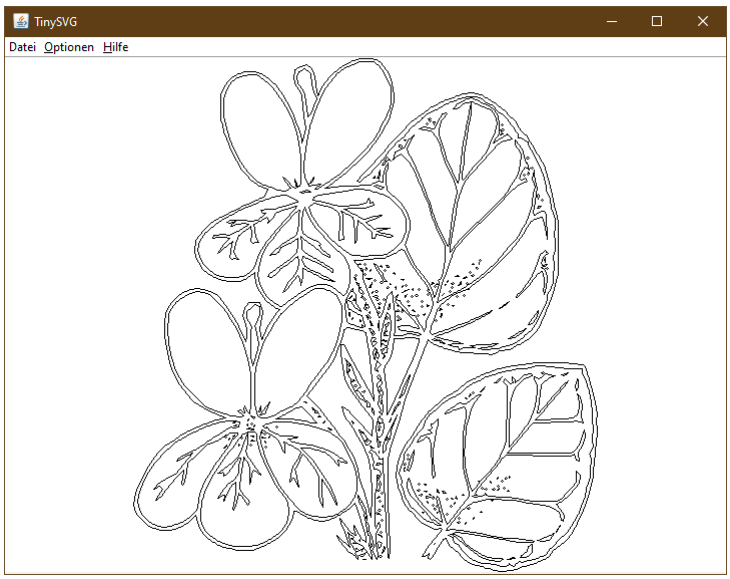
Konfigurations- und Log-Datei

Alle Optionen werden automatisch in einer Datei namens TinySVG.properties gespeichert, die im selben Verzeichnis wie TinySVG.jar abgelegt wird. Beim nächsten Start wird diese Datei gelesen und alle Optionen sind wieder so, wie man sie zuletzt eingestellt hat.



Daneben wird noch die Datei TinySVG.log erzeugt, die – wie die Dateiendung bereits vermuten lässt – das Log enthält. Über **Optionen > Log-Schweregrad setzen** kann man steuern, wie detailreich das Protokoll sein kann. Die Voreinstellung ist **Info**. Detailliertere Logs mit **Fine** oder **Finer** sind nur sinnvoll, wenn ein Fehler genauer analysiert werden muss. Die Log-Dateien werden dann aber deutlich größer und das Schreiben des Logs dauer länger. Man kann den Log-Schweregrad auch auf **Warning** oder **Severe** stellen, wenn man kürzere Logs bevorzugt.

Nachdem nun alle Optionen eingestellt sind, kann man eine SVG-Datei via **Datei > Öffnen** öffnen. TinySVG merkt sich den Pfad, den man zuletzt geöffnet hatte. Nachdem man die gewünschte Datei ausgewählt hat, wird die Datei umgewandelt und erscheint im Vorschaufenster. Auch die Ausgabedateien werden sofort erzeugt und in dem Verzeichnis gespeichert, in dem sich auch die SVG-Datei befindet.



Wenn man nach dem Laden einer Datei noch mit den Optionen experimentieren möchte (z.B. mit dem Bézier-Faktor), dann kann man die Änderungen bequem über **Datei > Neu laden** zur Anzeige bringen.

Mit **Datei > Beenden** wird das Programm beendet.

Unter **Hilfe** gibt es nur den Menüpunkt **Über**, mit dem ein Fenster angezeigt wird, das Auskunft über die TinySVG-Version und den Autor des Programms gibt.

Plattformen und deren Hilfsprogramme

Schneider Joyce

Die empfohlenen Einstellungen unter Ausgabedateien und Bildschirmauflösung sind **VE2 (.ve2)** und **720x256 (Schneider Joyce)**. Die erzeugten VE2-Dateien können mit dem Hilfsprogramm vecread auf der Joyce angezeigt werden. Das ältere Format VEC (.vec) kann zwar auch verwendet werden, ist aber nicht mehr zu empfehlen.

Alternativ können auch Dateien im HP-GL-Format ausgegeben werden: **HP-GL (.plt)**. Für die Anzeige der PLT-Dateien auf der Joyce steht der HP-GL Interpreter zur Verfügung.

Hilfsprogramm vecread

Das CP/M-Programm `vecread` kann Dateien im VE2- oder VEC-Format verarbeiten. Man kann das Programm auf der Joyce einfach durch Eingabe von `vecread` starten, woraufhin ein Menü erscheint, das u.a. zur Eingabe eines Dateinamens auffordert. Der Dateiname muss vollständig, also inklusive der Extension .ve2 oder .vec, eingegeben werden. Alternativ kann man das Programm auch mit der gewünschten Datei als Parameter aufrufen, also z.B. `vecread puffin.ve2`. Das Zeichnen der Grafik beginnt dann sofort. Ein Ton signalisiert das Ende des Zeichenvorgangs; ein Tastendruck beendet das Programm.

Hilfsprogramm HP-GL Interpreter

Der Dateiname des HP-GL Interpreters lautet HPGLxx.COM, wobei „xx“ für die Versionsnummer steht, also z.B. HPGL34.COM für Version 3.4. Das Programm wird durch Eingabe des Dateinamens gestartet, z.B. `HPGL34`. Es erscheint ein Menü, in dem man zunächst den Namen der HP-GL-Datei eingibt, z.B. **girl2.plt**.

Mit der Tab- oder der Enter-Taste gelangt man in das nächste Feld, in dem man einen Dateinamen eingeben kann. Lässt man das Feld leer, dann wird keine PBM-Datei erzeugt. Füllt man dieses Feld aus, z.B. mit girl2.pbm, dann wird eine PBM-Datei dieses Namens erzeugt. PBM steht für „**P**ortable **B**it**M**ap“, ein standardisiertes Bitmap-Format. PBM-Dateien lassen sich auf der Joyce z.B. mit dem Programm `pbmread` einlesen und anzeigen. Da die Pixel auf der Joyce etwa doppelt so hoch wie breit sind, kann mit bei „AR correction“ mit „Y“ bestätigen, woraufhin das Bild in der PBM-Datei doppelt so hoch wird und damit auf einem PC-Monitor unverzerrt dargestellt wird.

Wenn man nur eine HP-GL-Datei schnell anzeigen möchte, dann kann man den Dateinamen auch direkt als Kommandozeilenparameter angeben, z.B. `HPGL34 girl2.plt`. Das Bild wird dann sofort gezeichnet.

Sinclair ZX81

Für den ZX81 erzeugt TinySVG .p-Dateien, die die Vektordaten in einer REM-Zeile enthalten. Diese können direkt im Emulator ausgeführt werden. Um sie auf der echten Hardware auszuführen, muss man sie irgendwie in den Speicher des ZX81 laden, z.B. indem man sie auf eine SD-Karte für ein ZXpand kopiert und dort startet. Für die Darstellung der hochauflösenden Grafik ist HRG-ms von Mathias Swatosch erforderlich: http://www.swatosch.de/zx81/index.html

Man benötigt kein separates Anzeigeprogramm; das Programm in der .p-Datei stellt die Grafik selbst dar.

Einstellungen in EightyOne:

1. Options > Hardware bzw. F6

2. Register "Sinclair"
3. ZX81 wählen
4. RAM Pack: 48k einstellen
5. Register "Interfaces": High Resolution auf "WRX" setzen
6. Register "Advanced Settings": "Enable M1Not Circuit" aktivieren

Grafik laden in EightyOne:

1. Tape Manager via Tools > Tape Manager aktivieren
2. Im Tape Manager die "Auto LOAD on insert"-Schaltfläche deaktivieren (ganz rechts oben); "Flash Load" und "Auto Start/Stop" bleiben an
3. File > Open Tape > **hrg-64k.p** auswählen
4. LOAD "" ⇨ das HRG-ms Startbild erscheint
5. Leertaste drücken ⇨ HRG zeigt an, in welchen Speicherbereich es geladen wurde, z.B. "OK. HRG LOADED TO 30144"
6. Im Tape Manager auf die Öffnen-Schaltfläche klicken und die .p-Datei auswählen
7. Im EightyOne-Hauptfenster LOAD "" eingeben
8. RUN eingeben

Sinclair ZX Spectrum

Für den ZX Spectrum gibt es gleich drei Dateiformate: **ZX Spectrum standard (.zxv)**, **ZX Spectrum speedplot (.zxs)** und **ZX Spectrum speedplot extended (.zxx)**. Jedes der drei Formate hat ein zugehöriges Anzeigeprogramm, das jeweils als TZX-Datei vorliegt.

1. Standard bzw. Standard-Plot war der erste Ansatz, wobei die ins BASIC eingebauten Grafikbefehle PLOT und DRAW verwendet werden. Da die beiden unteren Zeilen nicht von diesen Befehlen erreicht werden, beträgt die Auflösung nur 256x176 Punkte. Die Zeichengeschwindigkeit ist eher langsam.
2. Speedplot ist ein Maschinencode-Programm, das Vektoren sehr schnell zeichnen kann. Das Dateiformat ist sehr einfach; es werden nur Linien unterstützt.
3. Speedplot extended unterstützt neben Linien auch weitere Grafikobjekte; daher ist das Dateiformat kompakter.

Standard-Plot im Emulator EightyOne:

1. File > Open Tape > **standard-plot.TZX** auswählen. Das Programm startet automatisch und setzt RAMTOP auf 29999.
2. File > Load Memory Block, dann unter **Filename** die gewünschte **.zxv**-Datei auswählen und unter **Address** 30000 eingeben. Dann auf **Load File** klicken.
3. RUN

Speedplot im Emulator EightyOne:

1. File > Open Tape > **speedplot_basic.tzx** laden. Das Programm startet automatisch und setzt den RAMTOP auf 29999.
2. File > Load Memory Block > Filename: speedplot.bin, Address: 65145 - das lädt den Maschinencode.
3. File > Load Memory Block > Filename: z.B. laempel.zxs, Address: 30000 - damit werden die von TinySVG erzeugten Vektordaten geladen.
4. RUN

Die alternative TZX-Datei speedplot_all.tzx lädt alles automatisch: zuerst die BASIC-Datei, dann wird RAMTOP geändert, und schließlich wird der speedplot-Maschinencode geladen. Man muss dann nur noch die Vektordaten laden und RUN eingeben:

1. File > Open Tape > **speedplot_all.tzx** laden. Das Programm startet automatisch, setzt den RAMTOP auf 29999 und lädt den Maschinencode.
2. File > Load Memory Block > Filename: z.B. laempel.**zxs**, Address: 30000 - damit werden die von TinySVG erzeugten Vektordaten geladen.
3. RUN

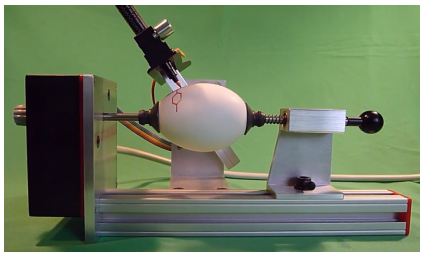
Speedplot extended im Emulator EightyOne:

1. File > Open Tape > **speedplot_ext.tzx** laden. Das Programm startet automatisch und setzt den RAMTOP auf 29999.
2. File > Load Memory Block > Filename: speedplot_ext.bin, Address: 64800 - das lädt den Maschinencode.
3. File > Load Memory Block > Filename: z.B. laempel.zxx, Address: 30000 - damit werden die von TinySVG erzeugten Vektordaten geladen.
4. RUN

Die alternative TZX-Datei speedplot_ext_all.tzx lädt alles automatisch: zuerst die BASIC-Datei, dann wird RAMTOP geändert, und schließlich wird der speedplot-extended-Maschinencode geladen. Man muss dann nur noch die Vektordaten laden und RUN eingeben:

1. File > Open Tape > **speedplot_ext_all.tzx** laden. Das Programm startet automatisch, setzt den RAMTOP auf 29999 und lädt den Maschinencode.
2. File > Load Memory Block > Filename: z.B. laempel.**zxx**, Address: 30000 - damit werden die von TinySVG erzeugten Vektordaten geladen.
3. RUN

ZX-Eiermaler



Der ZX-Eiermaler ist ein ZX81-gesteuerter Spezialplotter, der Eier (oder auch Tischtennisbälle o.ä.) bemalen kann, siehe https://forum.tlienhard.com/phpBB3/viewtopic.php?t=2278

TinySVG kann Steuerdateien für den ZX-Eiermaler erzeugen. Dafür gibt es zwei Optionen: ZX-Eiermaler (.bas) erzeugt eine BASIC-Datei, ZXCNC (.cnc) eine Datei mit Vektoren. Letztere ist zu bevorzugen, weil ZXCNC-Dateien kompakter sind und somit komplexere Grafiken ermöglichen.

In der TinySVG-Konfiguration sollte **Linienreduktion aktivieren** eingeschaltet sein. Damit werden zwei Arten von Linien, die unnötig sind und nur die Datei aufblähen und den Stift des Eiermalers malträtieren, entfernt:

- Linien der Länge Null, also eigentlich einzelne Punkte, die mit einer anderen Linie verbunden sind.
- Linien, die dieselbe Steigung (denselben Neigungswinkel) wie eine verbundene Linie haben. Um diese zu entfernen, muss die verbundene Linie entsprechend verlängert werden (aus zwei mach eins).

Hilfsprogramm PathOptimizer

Für den Eiermaler gibt es auch noch eine Wegeoptimierung (PathOptimizer) um die Stiftbewegungen auf ein Minimum zu reduzieren. Das Programm liest eine ZXCNC-Datei ein und macht eine Vorwärtsanalyse, die Streckenzüge nach definierten Regeln zusammenfasst. Ein *Streckenzug* ist dabei eine Liste verbundener Linien (genauer: Strecken). Streckenzüge werden anhand der Heben- und Senken-Funktion des Stifts erkannt.

C64

Hilfsprogramm DRAWVECTOR

Das C64-Programm DRAWVECTOR wurde in C unter Verwendung des cc65 Cross Compilers geschrieben und stellt die Vektorgrafiken zügig dar.

Das Programm kommt in einer D64-Datei, die man z.B. mit dem Emulator VICE öffnen kann. Es wird wie auf dem C64 üblich via `LOAD "DRAWVECTOR",8,1` geladen. Nach dem Start wird nach dem Dateinamen gefragt; man gibt z.B. laempel.ve2 ein. Alternativ kann das Programm mit „Q“ für „Quit“ beendet werden.

Wenn das Bild fertig gezeichnet wurde, kann man es betrachten, bis man durch einen Tastendruck in die Auswahl zurückkommt. Die Abfrage **again (y/n)** führt bei einem Tastendruck auf „Y“ zur erneuten Dateinamenseingabe; drückt man auf „N“, dann wird das Programm beendet.

Hilfsprogramm TsbTinyVECViewer

TsbTinyVECViewer wurde von GoDot aus dem Forum64 (https://www.forum64.de) entwickelt und läuft unter TSB (das ist eine fehlerbereinigte Version von Simons‘ BASIC). Es nutzt die Grafikbefehle des TSB, ist aber deutlich langsamer als DRAWVECTOR.

Zunächst lädt man TSB mit `LOAD "TSB",8,1`

Die Eingabe von RUN führt anschließend zum Booten von TSB. Wenn die ready-Meldung kommt, gibt man `LOAD "TSBTinyVECViewer",8` gefolgt von `RUN` ein. Die Dateien auf der Diskette werden angezeigt und man wird aufgefordert den gewünschten Dateinamen einzugeben.

Dann wird das Bild gezeichnet. Ein Druck auf eine Taste führt zur Meldung „ENTER startet neu. Ende mit 'x'.“ - das sollte selbsterklärend sein.

Bernd Bock, Oktober 2022