Grafik für Gforth

Hannes Teich

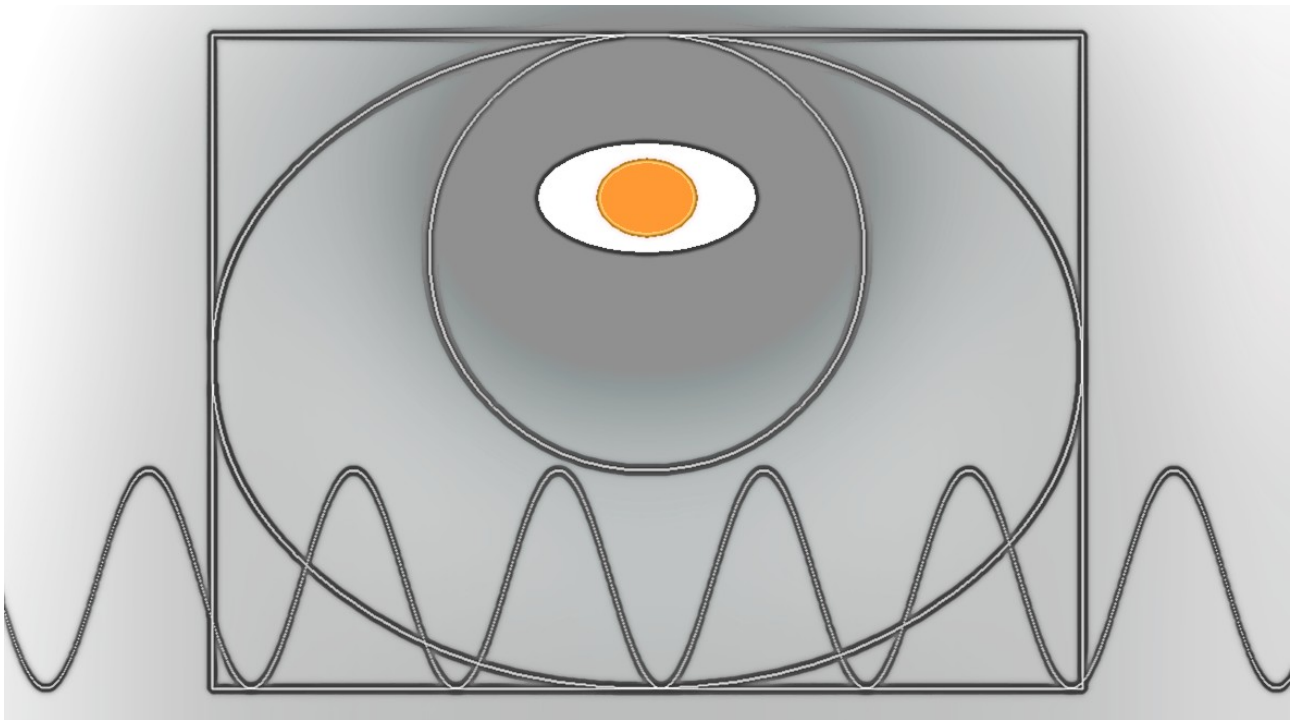

Abbildung 1: Grafik, mit Gforth erzeugt, mit Gimp verfremdet

Um es gleich vorweg zu sagen: Der Plotter ist ein Android-Tablet, in meinem Fall der Nexus-7 von Google/Asus. Die Verbindung dorthin übernimmt eine Datei.

Ich habe oft bedauert, dass Gforth keine Grafik-Schnittstelle bietet. Das ist seiner Portabilität geschuldet, denn jede Plattform will grafisch anders bedient werden. Bisher habe ich mir mit Basic-256 geholfen, das unter Linux und Windows läuft. Mit einer Bildschirmauflösung von 1820 x 1024 (Full HD) kriege ich damit 1200 x 666 als Grafikfenster.

Nun bin ich kürzlich in die Android-Welt eingestiegen, und weil ich dort zwar Gforth vorfinde, aber nicht Basic-256, so habe ich gesucht – und etwas noch Besseres gefunden, nämlich Mobile Basic, sehr tüchtig und gut dokumentiert, mit vielen Beispielen.

Ich habe mir ein Dateiformat ausgedacht und eine entsprechende Grafik-Datei zunächst mit Basic erzeugt, mit Basic gelesen und mit Basic auf den Bildschirm gezeichnet. Als das funktionierte, war es nicht mehr schwer, solch eine Datei mit Gforth zu erzeugen.

Als Android-Neuling habe ich mich schwer getan (und tue es noch), das Dateisystem zu durchschauen. Eine gute Übersicht habe ich noch nicht entdeckt. Deshalb ist vieles durch Versuch, Irrtum und Heureka zustande gekommen. Ob es elegantere Lösungen gibt, weiß ich nicht; ich beschreibe einfach, wie es bei mir funktioniert.

Gforth läuft auf Android im Ordner `/storage/emulated/0/gforth/site-forth/`, ist zwar auch anders adressierbar, aber das halbe Dutzend File-Manager, die ich mir geholt habe, halten `/storage/emulated/0/` für die gängige Ebene. Gforth legt seine Ausgabe-Dateien bevorzugt in `/storage/emulated/0/gforth/home/` ab, und deshalb

verwende ich diesen Ordner im Folgenden, obgleich ich den Weg vom Desktop zum Tablet schildern möchte. Die Tastatur des Desktop ist einfach leichter zu bedienen.

Es folgen zwei Listings, eines in Gforth, das andere in Mobile Basic geschrieben. Zusammen erzeugen sie das obige Bild (Abb. 1) vor der Gimp-Verfremdung.Vorher liste ich noch den Zyklus der Handgriffe auf, um von der Formulierung der Grafik bis zum Ergebnis zu kommen. Ich verwende mit Erfolg die zwei Apps: *wifi file transfer pro* von *smarterDroid* zur drahtlosen Datenübertragung sowie den *Android File Manager* von *SmartWho*.

– Plotdaten im Gforth-Programm bearbeiten;
– Terminal: `~/Desktop$ gforth plotter.fs` (plot.dat entsteht);
– Am Tablet `wifi file transfer pro` starten (`http://192.168.1.100:1234`);
– Browser am Desktop starten;
– Adresse `http://192.168.1.100:1234` im Adressfeld eingeben;
– Ordner `/storage/emulated/0/gforth/home/` wählen;
– Älteres `plot.dat` löschen (sonst wird neues umbenannt);
– *"Dateien auswählen"* anklicken;
– `plot.dat` im Desktop anwählen;
– *"Upload starten"* anklicken;
– Am Tablet `wifi file transfer pro` beenden;
– Mobile Basic aufrufen;
– `plotout.bas` starten (holt sich `plot.dat` aus home);
– Voila! - Die Grafik erscheint.

Und um das Ergebnis zu retten:
– Einen *Snapshot* machen (Austaste und Leiser-Taste zugleich);
– Den Plot in der *Galerie* aufsuchen;
– Wenn zufrieden, mit `wifi file transfer pro` zum Desktop holen;
– Dazu in `/storage/emulated/0/Pictures/Screenshots` nachsehen.

So, das war's. Weitere Informationen in den Listings.

... die wohl auch irgendwo bereitstehen zum Runterladen.

```
\ ===+=== 1 ===+=== 2 ===+=== 3 ===+=== 4 ===+=== 5 ===+=== 6 ===+===
\ plotter.fs - last edit: 19-feb-2014 21:00 -jgt
\
\ ##################################################################
\ Generieren einer durch Mobile Basic zu interpretierenden Plot-Datei
\ ##################################################################
\
\ Fuer die Grafik-Datei sind (vorerst nur) 10 Befehle vorgesehen, mit
\ denen in beliebigen Farben Linien, Kreise, Ellipsen, Rechtecke und
\ vor allem einzelne Pixel gesetzt werden koennen. Die Befehle werden
\ jeweils durch Kennnummern eingeleitet. Kennummern und Parameter
\ haben 16 bit Breite (Bytes-Anordnung: big endian). Fuer den Befehl
\ plot sind als Parameter beliebig viele xy-Paare erlaubt, die durch
\ eine Ende-Kennung abgeschlossen werden. clear fuellt den Bild-
\ schirm mit vorgewaehlter Farbe, show macht die gesetzte Grafik
\ sichtbar.
\
\   "_color"    | $AA01 |  <r>  |  <g>  |  <b>  |  <a>  |
\   "_clear"    | $AA02 |
\   "_line"     | $AA03 |  <x1> |  <y1> |  <x2> |  <y2> |
\   "_rect"     | $AA04 |  <x>  |  <y>  |  <w>  |  <h>  |  <f>  |
\   "_oval"     | $AA05 |  <x>  |  <y>  |  <w>  |  <h>  |  <f>  |
\   "_circle"   | $AA06 |  <x>  |  <y>  |  <r>  |  <f>  |
\   "_plot"     | $AA07 |  <x1> |  <y1> | . . . | . . . | $AA00 |
\   "_pause"    | $AA08 |  <s>  |
\   "_show"     | $AA09 |
\   "_end"      | $AA0A |
\
\ <r><g><b> = rot gruen blau (0 0 0 = schwarz, 255 255 255 = weiss)
\ <a> = alpha (0 = transparent, 255 = deckend)
\ <x><y><r> = Kreismittelpunkt und -radius
\ <x><y> = linke obere Ecke (Rechteck und Rahmen fuer Ellipse)
\ <w><h> = Breite und Hoehe (Rechteck und Rahmen fuer Ellipse)
\ <f> = ausfuellen (true/false fuer Kreis, Ellipse und Rechteck)
\ <s> = Sekunden (fuer Pause)

  s" deskplot.dat" 2constant  wfile       \ Dateiname
  0                value      wfileID     \ Dateikennung
  2000             constant   bufsize     \ Puffergroesse
                   variable   bufcnt      \ Pufferzaehler
                   variable   tmp         \ Fluechtige Variable

create wbuffer bufsize allot             \ Schreibpuffer

\ Ausgabedatei erzeugen
: create-wfile  ( --)
            wfile R/W BIN  ( c-addr u fam)
            CREATE-FILE      ( fileid ior)
       IF    ." couldn't create " wfile type ."  - quit" quit
       THEN  to wfileID ;

\ Daten in Wave-Datei schreiben.
: >file  ( addr len)  wfileID  WRITE-FILE ( ior) drop ;

\ Schreibpuffer leeren und in Datei schreiben.
\ FLUSH-FILE leert auch den unsichtbaren Zwischenspeicher.
\ Am Programmende ist der Puffer mit bufflush zu leeren.
: bufflush  ( --)
            wbuffer bufcnt @ >file
            0 bufcnt !
            wfileID FLUSH-FILE drop ;
```

```
\ Schreibpuffer laden und ggf. in die Wave-Datei uebertragen.
\ So lange die Daten nicht in den Puffer passen, wird der Puffer
\ randvoll gefuellt und sodann in die Wave-Datei entleert.
\ Dann wird der Datenrest in den Puffer geladen.
: $>buf  ( addr len --)
       BEGIN   bufsize bufcnt @ - >r
               dup r@ >=
       WHILE   over wbuffer bufcnt @ + r@ move
               swap r@ + swap  r> -
               wbuffer bufsize >file
               0 bufcnt !
       REPEAT  r> drop  dup
       IF      >r wbuffer bufcnt @ + r@ move
               bufcnt @ r> + bufcnt !
       ELSE    2drop
       THEN ;

\ 2 niederwertige Bytes vertauschen
: byte-swap  ( u1 -- u2 )  dup 8 rshift 255 and
                           swap 255 and 8 lshift or ;

\ 2-byte-Wert in den Schreibpuffer schreiben.(big endian)
: 2>buf  ( u --)  byte-swap  tmp !  tmp 2 $>buf ;


\ ##############################
\ Grafikdaten in Datei schreiben
\ ##############################

\ Dateibeginn
: _begin  ( -- )  cr ." begin "
       bufflush ;

\ Farbe waehlen
: _color  { r g b a -- }  ." color "
       $AA01 2>buf  r 2>buf  g 2>buf  b 2>buf  a 2>buf ;

\ Bildschirm loeschen
: _clear  ( -- )  ." clear "
       $AA02 2>buf ;

\ Gerade zeichnen
: _line  { x1 y1 x2 y2 -- )  ." line "
       $AA03 2>buf  x1 2>buf  y1 2>buf  x2 2>buf  y2 2>buf ;

\ Rechteck zeichnen (hohl oder gefuellt)
: _rect  { x y w h f -- }  ." rect "
       $AA04 2>buf  x 2>buf  y 2>buf  w 2>buf  h 2>buf  f 2>buf ;

\ Ellipse zeichnen (hohl oder gefuellt)
: _oval  { x y w h f -- }  ." oval "
       $AA05 2>buf  x 2>buf  y 2>buf  w 2>buf  h 2>buf  f 2>buf ;

\ Kreis zeichnen (hohl oder gefuellt)
: _circle  { x y r f -- }  ." circle "
       $AA06 2>buf  x 2>buf  y 2>buf  r 2>buf  f 2>buf ;

\ Ein oder mehrere Pixel setzen
: _plot-on   ( -- )        $AA07 2>buf  ." plot-on " ;
: _plot      { x y }  x 2>buf  y 2>buf  ." plot " ;
: _plot-off  ( -- )        $AA00 2>buf  ." plot-off " ;
```

```
\ Pause (Sekunden)
: _pause  { s -- }  ." pause "
       $AA08 2>buf  s 2>buf ;

\ Gesetzte Grafik sichtbar machen
: _show  ( -- )  ." show "
       $AA09 2>buf ;

\ Dateiende
: _end  ( -- )  ." end " cr cr
       $AA0A 2>buf  bufflush ;


\ ######################
\ Eine Grafik formulieren
\ ######################

  create-wfile                   \ Ausgabe-Datei erzeugen

: go   bufflush
                        _begin      \ Dateibeginn
    0   0   0 255       _color      \ schwarz deckend
                        _clear      \ Bildschirm fuellen
  255 255 255 255       _color      \ weiss deckend
  400 300 800 600   0   _rect       \ Rechteck
  400 300 800 600   0   _oval       \ Ellipsen
  800 500 200   0       _circle     \ Kreis
                        _show       \ Grafik aktualisieren
                    2   _pause      \ Pause 2 Sekunden
  700 400 200 100   1   _oval       \ Ellipse ausgefuellt
  255  40  40 127       _color      \
  750 410 100  80   1   _oval       \ Ellipse ausgefuellt
                        _show       \ Grafik aktualisieren
                    2   _pause      \ Pause 2 Sekunden
  255 180 180 255       _color      \ Rot fuer Plot
                        _plot-on    \ Plot
  1200 0 do i 200 + i s>f 30e f/ fsin 100e f* 800e f+ f>s
                        _plot       \ Sinuskurve
         loop           _plot-off
                        _show       \ Grafik aktualisieren
                   10   _pause      \ Pause 10 Sekunden
                        _end ;      \ Dateiende

  go    \ Autostart

\ ====================================================================
```

```
// ==+=== 1 ===+=== 2 ===+=== 3 ===+=== 4 ===+=== 5 ===+=== 6 ===+===
plotout.bas - Datei steuert Plotter (19-feb-2014 21:00 -jgt)

Dies ist ein Plotter-Programm fuer Mobile Basic unter Android.
Eine Datei mit Grafik-Befehlen steuert die Bildschirm-Grafik.
Die Datei plot.dat wird in /storage/emulated/0/gforth/home erwartet.
Dieser Ordner wird von Gforth fuer die Ausgabe benutzt.

Folgende Befehle werden erkannt:
    COLOR, CLEAR, LINE, RECT, OVAL, CIRCLE, PLOT, PAUSE, SHOW, END


function int(x as short) as short
    int=integer(x) & 0xFF
end function

sub gocolor     // {1} r g b a
    print "color"
    dim r,g,b,a as short
    get #1,r
    get #1,g
    get #1,b
    get #1,a
    setcolor r,g,b,a
end sub

sub goclear     // {2}
    print "clear"
    cls
end sub

sub goline      // {3} x1 y1 x2 y2
    print "line"
    dim x1,y1,x2,y2 as short
    get #1,x1
    get #1,y1
    get #1,x2
    get #1,y2
    drawline x1,y1,x2,y2
end sub

sub gorect      // {4} x y w h f
    print "rect"
    dim x,y,w,h,f as short
    get #1,x
    get #1,y
    get #1,w
    get #1,h
    get #1,f
    if integer(f)=0 then
        drawrect x,y,w,h
    else
        fillrect x,y,w,h
    end if
end sub

sub gooval      // {5} x y w h f
    print "oval"
    dim x,y,w,h,f as short
    get #1,x
    get #1,y
```

```
    get #1,w
    get #1,h
    get #1,f
    if integer(f)=0 then
        drawoval x,y,w,h
    else
        filloval x,y,w,h
    end if
end sub

sub gocircle    // {6} x y r f
    print "circle"
    dim x,y,r,f as short
    get #1,x
    get #1,y
    get #1,r
    get #1,f
    if integer(f)=0 then
        drawcircle x,y,r
    else
        fillcircle x,y,r
    end if
end sub

sub goplot      // {7} x1 y1 [x2 y2 [...]]
    print "plot"
    dim x,y as short
    dim endflag as boolean
    endflag=false
    while endflag=false
        get #1,x
        if x=short(0xAA00) then
            endflag=true
        else
            get #1,y
            plot x,y
        end if
    end while
end sub

sub gopause     // {8} s
    print "pause"
    dim s as short
    get #1,s
    sleep integer(s)*1000
end sub

sub goshow      // {9}
    print "show"
    repaint
end sub

sub main
    dim home,file as string
    dim ding as short
    dim item,cmd as integer
    dim basta,eflag as boolean

    graphics

    home="/storage/emulated/0/"
    file="gforth/home/deskplot.dat"
```

```
    open #1,home+file,"r"

    repeat
        repeat
            get #1,ding
            item=integer(ding)
        until item<0xAA00
        cmd=item & 0xFF
        if cmd=1 then
            call gocolor
        elseif cmd=2 then
            call goclear
        elseif cmd=3 then
            call goline
        elseif cmd=4 then
            call gorect
        elseif cmd=5 then
            call gooval
        elseif cmd=6 then
            call gocircle
        elseif cmd=7 then
            call goplot
        elseif cmd=8 then
            call gopause
        elseif cmd=9 then
            call goshow
        elseif cmd=10 then
            print "end"
            eflag=true
           end if
    until eflag=true

    close #1

 // basta=delete(home+file)  // basta=0 verschont die Datei
    if basta then
        print "file deleted"
    else
        print "file not deleted"
    end if
end sub

// ================================================================
```