B-VIEW

Großer Einsteiger-Teil: Der Umgang mit GOS/286

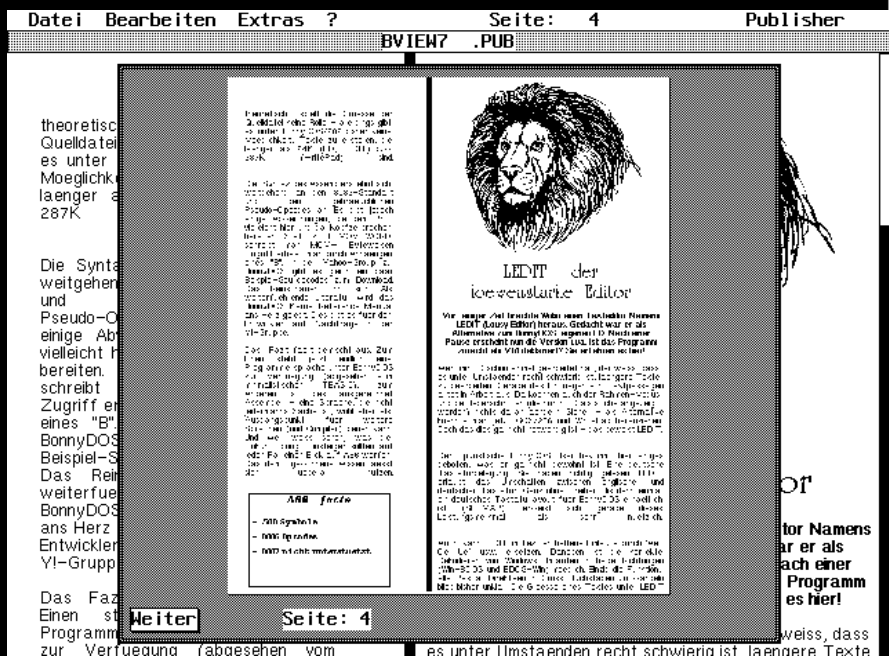
Neue Software: GOS-Paint im Test

Sonderteil Programmieren: Accessories im Selbstbau, die neue GOS286.TPU V2.0

....und vieles mehr!

```
BMPSAVER    Class: 10000000b, Location: $86D10
MOUSE       Class: 01000000b, Location: $87350
SMARTOPEN   Class: 01000000b, Location: $878B0
EQUAL       Class: 11111111b, Location: $884E0
SET         Class: 11111111b, Location: $885D0
DEVICE      Class: 11111111b, Location: $88AF0
EXIST       Class: 11111111b, Location: $89030
INFO        Class: 11111111b, Location: $890C0
RESIDENT    Class: 11111111b, Location: $89650
FASTDRIVE   Class: 01000000b, Location: $89910
EXTMEM      Class: 01000000b, Location: $8B310
NUL         Class: 01000001b, Location: $8BA70
LPT1        Class: 01000001b, Location: $8BA90
SEROUT      Class: 01000001b, Location: $8BB50
CRT         Class: 01000001b, Location: $8BBB0
KBD         Class: 01000010b, Location: $8BBD0

16 resident Programs/Drivers detected.
```




```
 Contents of Drive:

ADDON      .        --AD----      0K   AUTOBOOT.FD
AUTOBOOT.HD         --A-----      1K   AUTOBOOT.SCR
GOS        .        --AD----      0K   SHELL    .APL
SYSTEM     .        --AD----      0K   TUTORIAL.
DEBUG      .APL     --A-----     46K

3 hidden object(s),
5 file(s) and 4 dir(s)
64791 Block(s), 1012M free.

DEVC:
```

Editorial

Hoppla – da ist ja schon wieder eine Ausgabe der B-View! Nicht ganz! Diesmal halten Sie sozusagen ein Sonderheft in den Händen. Wozu das Ganze? Diese Ausgabe richtet sich an alle Einsteiger, die sich bisher noch nicht an BonnyDOS/286 oder GOS/286 herangetraut haben.

Das Handbuch hilft hier zwar weiter – doch geht es nicht auf alle möglichen Probleme und Hindernisse ein. Genau hier setzt diese Ausgabe der B-View an. Wer also neues Futter (Software) für seinen PC erwartet hat – der muss wohl oder übel auf die nächste, reguläre, Ausgabe der B-View warten.

Quasi etwas ganz Neues wurde bei der Erstellung dieser Ausgabe versucht. Statt den GOS-Publisher zu verwenden, wird der vorhandene Windows-PC und Open Office genutzt. Kritik, Anregungen oder Lob hierzu sind natürlich willkommen. Die Kontaktadresse finden Sie wie gewohnt auf http://lulu423gina.funpic.de.

Doch jetzt heißt es wieder: Viel Spaß beim Lesen dieser Ausgabe!

Ein Blick über den Tellerrand

GOS/286 für Einsteiger

Programme, die einheitlich aussehen und sich leicht per Maus bedienen lassen – das war der Hintergedanke bei der Einführung des Graphical Operating Systems, kurz GOS genannt. Wer sich bisher gescheut hat, einmal einen Blick darauf zu werfen, der kann sich in diesem kleinen Special ein Bild von diesem Softwarepaket machen.

Die Systemvoraussetzungen sind – anders, als von BonnyDOS gewohnt – relativ hoch gesteckt. Ein schneller 80286-Prozessor mit mindestens 16 Mhz sollte es schon sein. Unter BonnyDOS/286 sollten mind. 500K Konventioneller Speicher frei sein und – um alle Möglichkeiten nutzen zu können – nochmals mindestens 384K erweiterter Speicher installiert sein. Die restlichen Anforderungen sind dagegen nicht ungewöhnlich. So verlangt GOS/286 noch nach einer Maus (seriell oder PS/2), einer MCGA-/VGA-Karte, einem 3.5“-Laufwerk mit 1.44 MB Kapazität oder, noch besser, eine Festplatte.

Damit keine falschen Hoffnungen entstehen: Unter MS-DOS läuft GOS/286 freilich nicht. Der Grund ist schnell herausgefunden. Ein richtiges Betriebssystem ist GOS nämlich nicht. Vielmehr stellt es Programmen verschiedene Funktionen für den Betrieb im Grafikmodus zur Verfügung. Nach wie vor ist es aber – zum Beispiel bei Dateioperationen – auf BonnyDOS/286 angewiesen. Ein ähnliches Gespann kennt der PC-User bereits – die Rede ist von MS-DOS und Windows.

Langsam, langsamer – Diskette

GOS/286 kann kostenlos auf der BonnyDOS/286-Homepage oder in der Y!-Gruppe „bonnydos286“ heruntergeladen werden. Legen Sie sich außerdem noch drei formatierte 3.5“-HD Disketten zurecht. Kopieren Sie den Inhalt der jeweiligen Verzeichnisse („Disk1“,“Disk2“,“Disk3“) auf die entsprechende Diskette. Es reicht nicht, einfach das Directory auf die Diskette zu ziehen – Sie kommen nicht drum herum, wirklich alle Files in das Hauptverzeichnis jeder Diskette zu schieben.

Zu guter Letzt starten Sie noch BOOTGEN.EXE (MS-DOS/Windows), das sich nun auf Ihrer ersten Diskette befinden sollte. Von jetzt an ist Ihre GOS-Systemdiskette bootbar. Genau das sollten Sie jetzt machen. Starten Sie Ihren PC einmal neu und lassen Sie die Diskette 1 im Laufwerk. Nach wenigen Augenblicken erscheint das GOS/286-Startmenü. Hier können Sie mit den Tasten „1“ bis „3“ die installierte Maus auswählen. Hat Ihr Rechner nur bis zu 1 MB Arbeitsspeicher, müssen Sie wohl oder übel von Diskette aus starten. Drücken Sie dazu die Taste „5“ - sobald die Eingabeaufforderung erscheint, tippen Sie `GOS` ein und betätigen die `ENTER`-Taste. Beachten Sie, dass der Betrieb von Diskette nicht gerade für Freude sorgt. Das häufige Nachladen und die damit verbundene Wartezeit kann schnell für Frust sorgen.

Haben Sie mindestens zwei Megabyte RAM, empfiehlt es sich, nach Auswahl Ihrer Maus, die Taste 4 zu drücken. Der Computer aktiviert nun die BonnyDOS-eigene RAMDisk DEVB: und kopiert alle relevanten Dateien hinein. Anschließend startet GOS wie von Geisterhand in atemberaubender Geschwindigkeit. Zum Ausprobieren und Hineinschnuppern genau das Richtige. Gehören Sie zu dem erlauchten Kreis der Anwender, die BonnyDOS bereits auf Festplatte installiert haben, so spricht nichts dagegen, im Startmenü die Taste „6“ zu drücken (vorher noch die Maus auswählen!). Folgen Sie nun den Anweisungen am Bildschirm.

Der Schreibtisch

Nach einer mehr oder weniger kurzen Ladezeit erscheint der Schreibtisch. Über diesen lassen sich GOS- aber auch BonnyDOS-Programme starten. Das Bildschirmfoto zeigt einen schon recht gefüllten Bildschirm. Nach dem ersten Start von GOS/286 sieht das noch ganz anders aus – fast schon gähnende Leere herrscht hier am Monitor. Benutzen Sie zum Ausprobieren eine Diskette, so sorgen Sie dafür, dass die Schreibschutzöffnung geschlossen ist – das Schreiben also möglich ist. Bringen Sie außerdem viel Geduld mit – GOS/286 ist wirklich kein Geschwindigkeitswunder, wenn es von FloppyDisk geladen wird.

Wir starten ein Programm

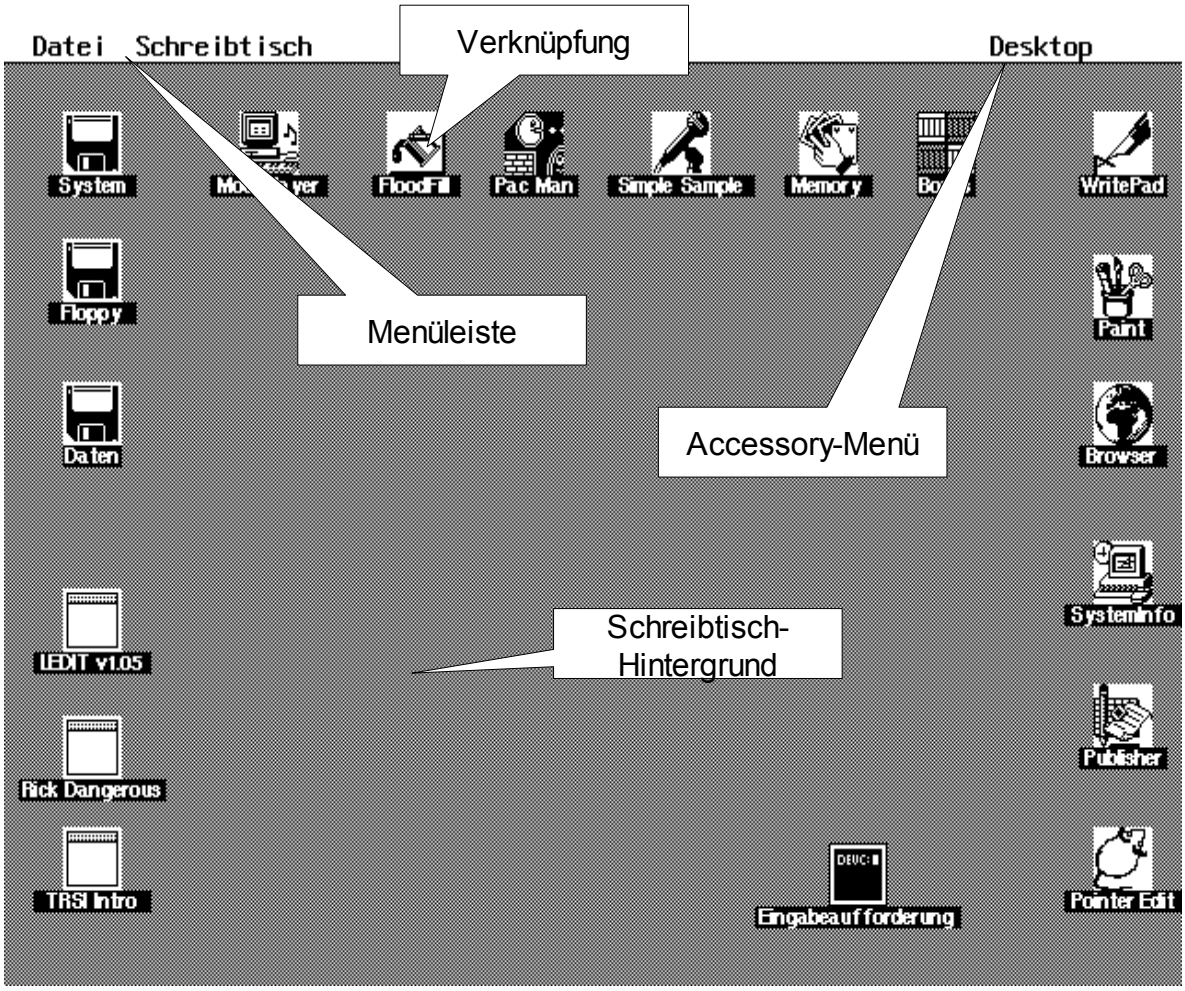
Es gibt unter GOS/286 zwei Möglichkeiten, ein Programm zu starten. Die erste ist das Doppelklicken auf eine Schreibtisch-Verknüpfung. Der zweite Weg ist das Menü „Datei/Ausführen“ (Tastenkürzel SHIFT+E). Im Gegensatz zu vorherigen GOS-Versionen ist es nun bei beiden Methoden möglich, BonnyDOS/286-Programme auszuführen. Beachten Sie aber, dass unter GOS/286 weniger Speicher frei ist und daher

Software, die auf viel RAM angewiesen ist, nicht lauffähig ist. Probieren Sie doch nun einmal beide Arten, Programme zu starten. Testen Sie, ob Sie das Programm WritePad („`WRITEPAD.APL`“) ausführen können. Ein Tipp: Writepad kann mit dem Menüpunkt „Datei/Beenden“ verlassen werden. Haben Sie Ihre erste Neugier befriedigt? Dann sollten Sie schnell weiter lesen. Bevor wir richtig einsteigen, heißt es nämlich erst einmal Grundlagen zu pauken. Doch keine Sorge – dem geübten PC-Anwender wird dies bereits von anderen Betriebssystemen bekannt sein.

Ein typisches GOS-Programm....

...sieht eigentlich immer gleich aus. Am oberen Bildrand wird die Menüleiste dargestellt. Das besondere hierbei ist bei der aktuellen GOS-Version das Accessory-Menü – vorerst reicht es zu wissen, das hierüber Tools (z. B. eine Uhr) geladen werden können. Im Falle des Schreibtisches werden am Bildschirm noch die Verknüpfungen und der Schreibtisch-Hintergrund dargestellt. Menüs lassen sich durch das Anklicken eines Menünamens aufklappen. Unter Anklicken versteht man dabei, den Mauspfeil auf die entsprechende Stelle am Bildschirm zu fahren und den linken Mausknopf zu drücken (nach dem Drücken gleich wieder loslassen!). Einen Menüpunkt wählen Sie auf die gleiche Weise aus. Natürlich wäre es fatal, wenn GOS bei einem aufgeklappten Menü immer auf eine Auswahl bestehen würde.

Daher lässt sich ein Menü auch wieder schließen, ohne einen Menüpunkt anzuklicken. Dazu bewegen Sie die Maus einfach irgendwohin, außerhalb des Menüs. Anschließend klicken Sie einmal mit der linken Maustaste auf diese Stelle. Voila! Das Menü verschwindet.

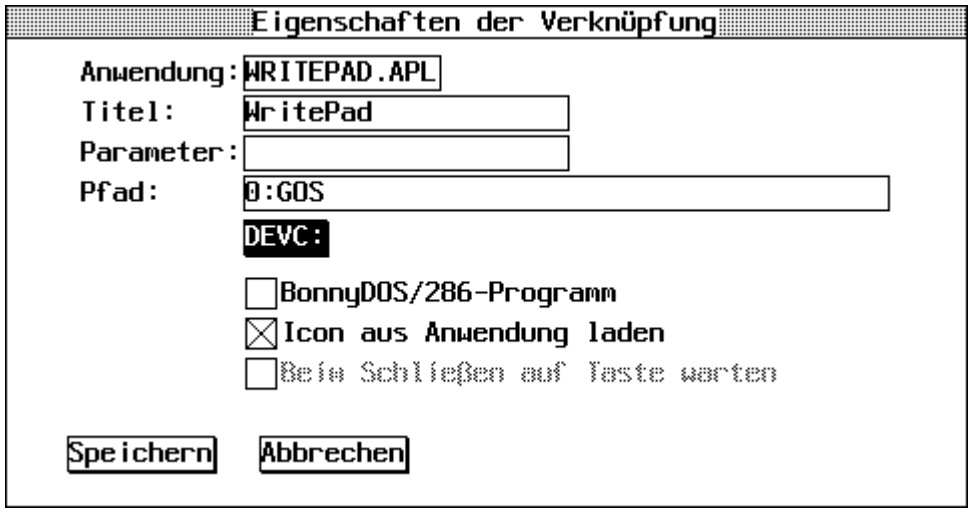


Das neu erworbene Wissen werden wir gleich einmal auf die Probe stellen. Wir werden uns nämlich gleich einmal eine Verknüpfung für ein Programm anlegen. Öffnen Sie dazu das Menü „Schreibtisch“. Klicken Sie jetzt den ersten Menüpunkt „Neue Verknüpfung“ an. Es öffnet sich nun der GOS-Dateidialog. Dieser wird Ihnen sicher noch öfters bei Ihrer Arbeit mit GOS-Programmen begegnen. Wir müssen uns nun überlegen, wie wir unsere Verknüpfung nennen – maximal 8

Buchstaben stehen zur Verfügung (siehe BonnyDOS-Handbuch „Dateinamen“). Klicken Sie mit der Maus bitte einmal in das Textfeld „Dateiname:“. Der Pfeil verschwindet nun – keine Bange! Sie befinden sich nun im Texteingabefeld. Drücken Sie jetzt bitte die Taste „ENTF“ (Entfernen). Das Feld wird dadurch geleert. Fürs erste nehmen wir einen einfachen Namen – tippen Sie also TEST ein. Drücken Sie ENTER oder ESC, um das Eingabefeld zu verlassen. Anschließend klicken Sie auf „Ok“. Ach ja – der Schreibtisch merkt (im Gegensatz zu seinen Vorgängern), dass der Zusatz „.LNK“ fehlt, oder falsch angegeben wurde. Die Korrektur wird ohne Ihr Wissen automatisch vorgenommen.

Ging alles glatt, dann sollte jetzt der Dialog für die Eigenschaften einer Verknüpfung erscheinen. Da Sie jetzt bereits mit Texteingabefeldern geübt sind, dürfte das Ausfüllen der selben nicht schwer fallen. Unter „Anwendung:“ tragen Sie den Dateinamen (mit oder ohne Zusatz „.APL“) ein. Wir geben für unseren Versuch einfach `GOSPAINT` ein. Den Inhalt im Feld „Titel:“ können wir frei wählen – hier sind auch Leerschritte erlaubt. Der hier eingegebene Text erscheint später unter dem Piktogramm.

Festplatten-Benutzer geben bei Pfad bitte `0:GOS` ein. Alle anderen lassen das Feld einfach leer. Die Schaltfläche unterhalb des Pfades klicken wir solange an, bis das gewünschte Laufwerk erscheint, auf dem sich das Programm befindet. Für RAMDisk-Nutzer heißt das richtige Laufwerk „DEVB:“, bei Diskette „DEVA:“ und bei Festplatte „DEVC:“. Was es mit den drei ankreuzbaren Punkten auf sich hat, das klären wir sofort.



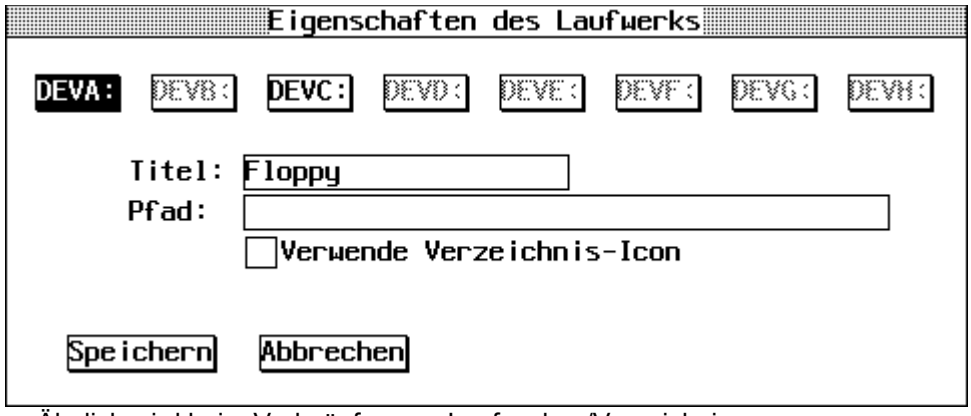
Die Eigenschaften einer Verknüpfung lassen sich hier beeinflussen

Da GOSPAINT kein BonnyDOS-Programm ist, muss das Kreuz unter „BonnyDOS/286-Programm“ entsprechend wegfallen. Diskettenbenutzer sollten jetzt aufpassen. GOS-Programme besitzen in der Regel ein eigenes Piktogramm, dass sich in der Programmdatei befindet. Kreuzen Sie „Icon aus Anwendung laden“ an, so versucht der Schreibtisch, dieses Piktogramm zu laden. Gerade hier ergeben sich beim Arbeiten mit Disketten Geschwindigkeitseinbußen. Deaktivieren Sie diesen Punkt ggf. - es wird dann ein Vorgabepiktogramm verwendet, welches nicht nachgeladen wird.

Der letzte Punkt „Beim Schließen auf Taste warten“ kann nur dann verwendet werden, wenn BonnyDOS/286-Programme gestartet werden sollen. Bevor GOS/286 beim Beenden solcher Software wieder den Schreibtisch einliest, wartet es, bis der Anwender eine Taste gedrückt hat. Dies kann immer dann nützlich sein, wenn ein Programm diverse Meldungen ausgibt, bevor es sich beendet. So bleibt genug Zeit, diese in Ruhe zu lesen.

Für unsere Testverknüpfung reicht es also zusammenfassend, wenn wir lediglich „Icon aus Anwendung laden“ aktivieren. Nun klicken wir auf die Schaltfläche „Speichern“. Das wars. Oben links am Schreibtisch sehen wir nun (hoffentlich) unsere erste Verknüpfung. Klicken Sie diese einmal an und lassen dabei die linke Maustaste gedrückt – wenn Sie nun die Maus bewegen, so verschiebt sich gleichzeitig das Piktogramm. An der gewünschten Stelle lassen Sie die Maustaste los. Die Positionen aller Verknüpfungen können Sie übrigens unter „Schreibtisch/Fixieren“ abspeichern.

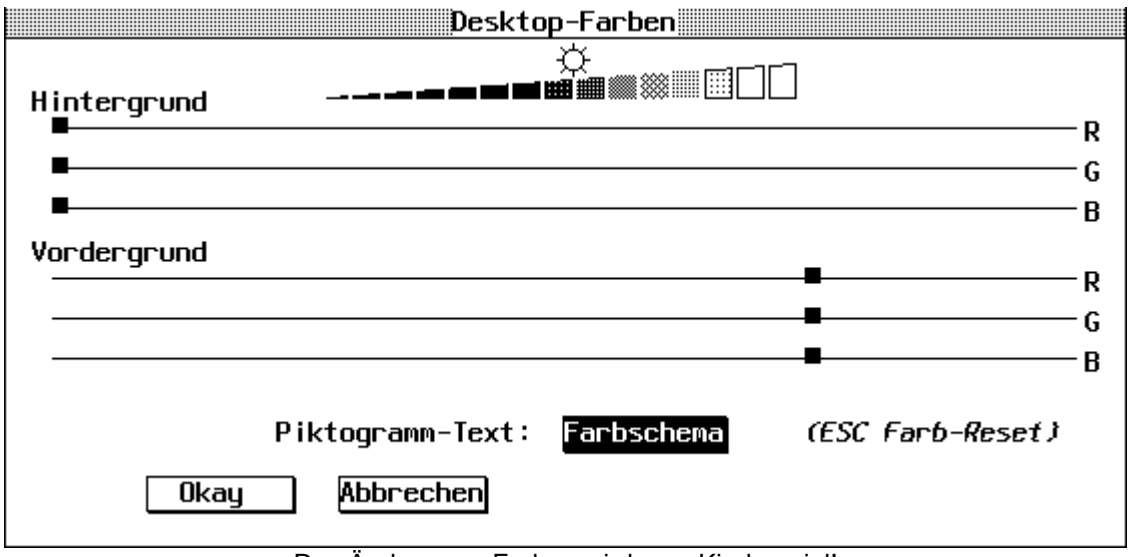
Sie können auch Laufwerke und Verzeichnisse als Verknüpfung anlegen – Beachten Sie jedoch, dass sich diese in der aktuellen GOS-Version noch nicht öffnen lassen, da der Dateimanager noch nicht fertiggestellt wurde. Das Anlegen solcher „Links“ verläuft ähnlich wie das eben geschilderte Verfahren. Lediglich der Dialog sieht ein wenig anderes aus. Außerdem müssen Sie darauf achten, dass das Textfeld „Pfad:“ auf Diskette und RamDisk gesperrt – also nicht zugänglich – ist. Experimentieren Sie ruhig ein wenig. Sie können nichts kaputt machen. Vor allen wichtigen Aktionen werden Sie aufgefordert, diese zu bestätigen. Sie haben also immer die Möglichkeit, bei auftretender Unsicherheit, einen Vorgang abzubrechen.



Ähnlich wird beim Verknüpfen von Laufwerken/Verzeichnissen vorgegangen

Farben und Hintergrund

GOS wäre nicht GOS, wenn Sie nicht auch die Möglichkeit hätten, beliebige Farbkombinationen zu verwenden und den Schreibtischhintergrund an Ihre Bedürfnisse anzupassen. Beginnen wir mit den Farben. Um diese zu ändern, wählen Sie im Menü „Schreibtisch“ den Punkt „Farben“ an. Für jeweils Vorder- und Hintergrundfarbe lassen sich hier Farben mischen. Klicken Sie dazu einfach auf eine Stelle in den Schiebereglern, um den jeweiligen Farbanteil („R“ot, „G“rün, „B“lau) zu ändern. Sind Sie in die peinliche Lage gekommen, dass die Farben so ungünstig gewählt wurden (zum Beispiel komplett Schwarz)? Kein Problem – Drücken Sie in diesem Fall die ESC-Taste und schon erhalten Sie die Vorgabe-Palette (schwarz/weiß). Im Farbdialog können Sie auch bestimmen, ob der Piktogrammtext von Verknüpfungen schwarz auf weiß oder umgekehrt dargestellt werden soll. Klicken Sie dazu einfach auf die Schaltfläche „Farbschema“. So, wie die Beschriftung des Buttons dargestellt wird, so erscheint später auch der Piktogrammtext. Zum Übernehmen Ihrer Komposition klicken Sie einfach auf „Okay“. Auf ähnlich einfache Weise kann der Schreibtischhintergrund angepasst werden.



Das Ändern von Farben wird zum Kinderspiel!

Der Hintergrund-Dialog

Sie können eines der vorhandenen Muster, oder ein unkomprimiertes Windows-BMP-Bild im Format 640x480x1 Bit verwenden. Den Namen des Bildes geben Sie entweder direkt im Texteingabefeld ein, oder Sie lassen sich den bereits bekannten Datei-Dialog durch Anklicken der Schaltfläche „...“ öffnen. Hier werden Ihnen alle vorhandenen Bitmap-Bilder angezeigt. Ausschlaggebend, ob Schreibtisch das Bild wirklich anzeigt, ist das Kreuz bei „Verwende Bild“. Durch Klick auf „Okay“ können Sie Ihre Einstellungen gleich einmal testen. Doch aufgepasst! Wenn Sie zwischenzeitlich ein Programm ausführen, ohne Ihre Farb-/Hintergrundeinstellungen zu speichern (Menü „Schreibtisch/Konfig sichern“), dann wird nach dem Wiedereinladen des Schreibtisches die zuletzt gesicherte Konfiguration verwendet. Denken Sie also daran, bei Bedarf immer gleich zu speichern.

Die Programme

Sie haben jetzt genug Grundlagen vorgesetzt bekommen. Jetzt wird es Zeit, einmal die zu GOS/286 gehörenden Programme unter die Lupe zu nehmen. Alle Programme (mit Ausnahme der beiden Spiele Pac4GOS und Memory) befinden sich auf der ersten Diskette (oder eben in der RAMDisk). Festplatten-Benutzer, die GOS/286 vollständig installiert haben, können sich bequem zurück lehnen – im Verzeichnis 0:GOS befindet sich das Komplettpaket.

GOS-Paint

Brandneu ist GOS-Paint, ein Malprogramm, das die hohe Auflösung der MCGA-/VGA-Karte nutzt. Eine ausführliche Beschreibung finden Sie im Anschluss an diesen Einsteigerteil. Interessierte sollten einfach einen Blick darauf werfen.

Writepad

Hierbei handelt es sich um einen kleinen Texteditor, der immerhin bis zu 287K Textspeicher zur Verfügung stellt. Texte lassen sich importieren und als ASCII-File exportieren. Ebenfalls kann der Anwender zwischen dem IBM- und CPC-Zeichensatz umschalten. Als kleines Bonbon können Windows BMP-Bilder in den Text eingefügt werden. Möchten Sie nicht, das andere Leute Ihre Texte bearbeiten? Dann können Sie das Dokument mit einem Passwort versehen. Es lässt sich dann zwar weiterhin lesen – jedoch nicht mehr Ändern.

Browser

Aktuell handelt es sich hier um einen Offline-Browser zur Darstellung von IGF-Dateien (Interactive Guide Format). Es handelt sich dabei um eine Art von HTML-Steuercodes, mit denen die Darstellung beeinflusst werden kann. Wie es sich für ein gutes GOS-Programm gehört, lassen sich z. B. verschiedene Schriftarten und auch BMP-Bilder nutzen. Einige Ausgaben der B-View sind zum Beispiel als IGF-Version erschienen.

Pointer Editor

Gerade für Programmierer ist dieses kleine Tool nützlich. Es lassen sich so eigene Mauszeiger erstellen, die Sie später in Ihrer eigenen Software nutzen können.

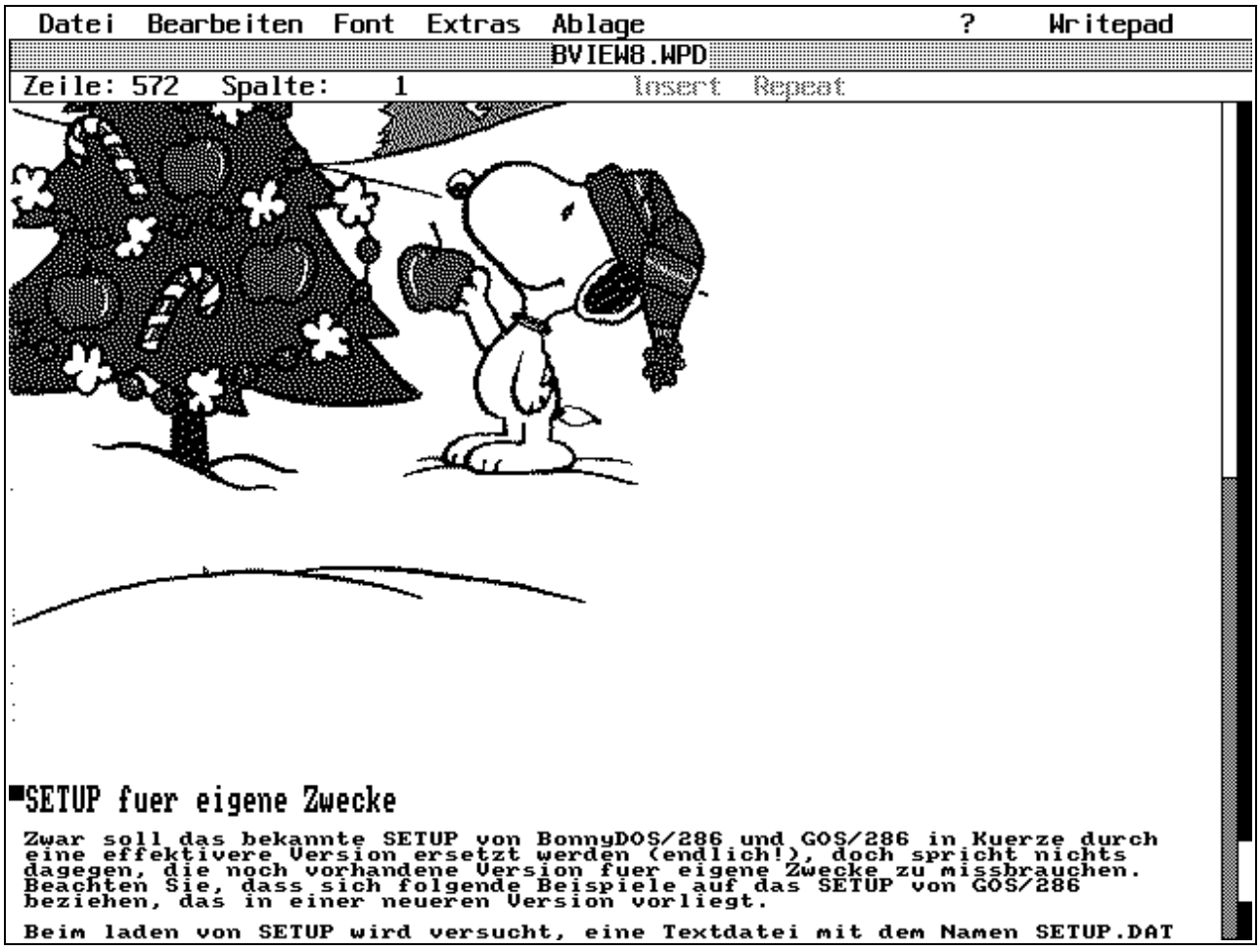
Systeminfo

Das Programm zeigt Ihnen die verwendete Version von GOS, dem Grafiktreiber und von BonnyDOS an. Bevor Sie Ihr System aktualisieren, können Sie nachsehen, welche Systemdateien Sie bereits benutzen.

Spiele, Spiele, Spiele

Zu GOS/286 gehören auch drei Spiele. Bei Boxes handelt es sich um einen waschechten Sokoban-Clone mit 50 Levels. Fortschritte können natürlich gespeichert und wieder eingeladen werden. Wer seine Hirnwindungen trainieren möchte, der ist bei Boxes genau richtig. Die beiden anderen Spiele

befinden sich auf der dritten GOS-Diskette. Pac4GOS wurde von Wobo geschrieben. Es ist eine ausgezeichnete Umsetzung des bekannten Klassikers Pac Man. Hinzugekommen sind jedoch einige Extras, die das Leben des Spielers schwer machen. Memory richtet sich wieder an die Knobelfreunde. Unter drei verschiedenen Schwierigkeitsstufen darf man sein Gedächtnis mit dem des Computers messen. Finden Sie mehr Kartenpaare als Ihr Gegner?



Writepad – der Texteditor für GOS/286

Das Programmieren von Accessories

Zubehör im Selbstbau

Zwar wurde zu diesem Thema bereits ein Dokument veröffentlicht, doch hat dies durch die neuen Möglichkeiten von GOS fast schon seine Gültigkeit verloren. Programmierer sollten also diesen Artikel aufmerksam durchlesen, wenn die Selbstbau-Accessories reibungslos funktionieren sollen.

Ein Accessory ist äußerlich an seinem Dateinamen erkennbar. Dieser trägt nämlich immer den Zusatz „.ACC“. Im Inneren sind Accessories fast identisch mit normalen GOS-Anwendungen. Aber eben nur fast! Während ein GOS-Programm beim Start immer die selben Bedingungen (Bildschirm, Maus, usw.) vorfindet, ist dies beim Zubehör ganz anders. Es wird nämlich aus einer laufenden Anwendung heraus gestartet. Dazu „friert“ GOS/286 die aktive Anwendung einfach ein und sichert diese im Speicher oberhalb von 1 MB. Anschließend wird das Accessory geladen und ausgeführt.

Alleine schon aus diesem Ablauf heraus wird ersichtlich, dass ein Accessory sich um einige Dinge selbst kümmern muss, um nicht die Arbeit mit dem ursprünglichen Programm zu stören. Zum wichtigsten Punkt gehört das sichern und wiederherstellen der Bereiche des Bildschirms, die vom Zubehör genutzt werden. Ausgenommen sind hier natürlich Systemdialoge (z. B. bei CLOCK.ACC) – werden ausschließlich solche verwendet, müssen weder beim Start, noch beim Beenden bestimmte Vorkehrungen getroffen werden.

Achten Sie immer auf den aktuellen Viewport (das aktive Grafikfenster) und auf einen ggf. aktiven Mausinterrupt, wie er erstmals von GOS-Paint verwendet wird. Deaktivieren Sie diesen eventuell während des Ablaufs des Zubehörs. Beim Beenden sollten Sie alles wieder so hinterlassen, wie Sie es vorgefunden haben. Und hier muss besondere Sorgfalt angewendet werden. Bei eigenen Dialogen beispielsweise muss die Funktion PermitACC von GOS genutzt werden, da der Anwender sonst ein weiteres Accessory laden könnte – der Rechner stürzt dann hoffnungslos ab!

Beachten Sie, dass Schreibtischzubehör zwar kein vollwertiger Ersatz für den alten Task-Switcher ist, jedoch – sinnvoll angewendet – dessen Verlust gut verschmerzen lässt. Bevor Sie ein selbstgeschriebenes Accessory freigeben, sollten Sie es mit diversen GOS-Programmen einmal testen. Erst wenn alles klappt, kann die Datei verteilt werden. Achten Sie beim Test auch immer auf Ihre GOS-Version – sind Sie nicht auf dem Laufenden, können schnell Probleme entstehen. Erst wenn Sie alle hier aufgeführten Artikel beherzigen, steht dem Erfolgreichen Programmieren von ACCs nichts im Wege.

```
Um Accessories nutzen zu können, muss
Ihr Rechner über mindestens 1 MB
Arbeitsspeicher verfügen. Zum
Zeitpunkt des Starts von Zubehör
müssen davon 384K frei sein.
Ebenfalls ist es erforderlich, vor
dem Start von GOS/286 EXTMEM zu
laden.

Alle im System installierten
Accessories werden beim Start von
GOS/286 ins Accessory-Menü
eingetragen. Dieses öffnet sich durch
Anklicken des Namens der laufenden
Anwendung (in der Menüleiste, rechts
am Bildrand.
```

Malen nach Zahlen

GOS-Paint stellt sich vor

Frisch vom Entwickler-Rechner kommt das neueste Familienmitglied der GOS/286-Familie. GOS-Paint ermöglicht es, seiner kreativen Ader freien Lauf zu lassen und verwandelt den Monitor ein ein elektronisches Zeichenblatt. Ob das Programm dabei eine gute Figur macht?

Um effektiv mit GOS-Paint zu arbeiten, sollte Ihr Rechner über einen Megabyte Arbeitsspeicher verfügen. Ebenfalls wird ein flotter 80286-Prozessor zwingend vorausgesetzt – ansonsten heißt es beim Malen Kaffee kochen. Wie fast alle neuen GOS-Programme wurde GOS-Paint mit Turbo Pascal entwickelt. Dort, wo es auf Geschwindigkeit ankommt, wird der Rechner durch handgeschriebenen Maschinencode auf Trab gebracht. Extra für GOS-Paint wurde auch der Grafiktreiber GRAPHIC.DRV überarbeitet – erst so wurden die ganzen Zeichenfunktionen erst möglich.

Leistung ohne Grenzen – oder Grenzen der Leistung?

Neben acht Punktstärken, 23 Mustern (Zeichenfarben), 8 selbstdefinierbaren Mustern und allen gängigen Zeichenfunktionen (Freihand, Linie, Kreis/Ellipse, Füllen) gibt es auch eine Textfunktion, ein Vergrößerungsglas und eine Funktion zum Speichern von Bildschirm-Ausschnitten. Herz, was willst du mehr! Während die Zeichenfunktionen über Schaltflächen am unteren Bildrand aktiviert werden können, wird alles andere über Pulldownmenüs oder Tastenkürzel erledigt. Schön anzusehen ist, dass GOS-Paint vom neuen Grafiktreiber seinen Nutzen zieht. Der Anwender sieht zum Beispiel schon beim Aufklappen der Menüs, welche Füllmuster und Punktstärken zur Verfügung stehen. Ein lästiges Herumprobieren entfällt somit.

Wo Licht ist, ist auch Schatten...

Negativ fallen einige Zeichenfunktionen auf. So werden Scheiben (gefüllte Kreise/Ellipsen) und Blöcke (gefüllte Rechtecke) recht träge gefüllt. Hier bedarf es auf jeden Fall einer Überarbeitung. Etwas säuerlich stößt auch die fehlende Undo-Funktion auf. Bis es diese gibt, sollte man vor allen wichtigen Schritten das Speichern nicht vergessen. Besonders überzeugen konnte die Textfunktion (siehe Bild). Diese macht regen Gebrauch von den verschiedensten Schriftarten – eine Festplatte macht sich hier richtig bezahlt.

Die Textfunktion

Die Lupenfunktion erlaubt es, einen beliebigen Bildausschnitt zu vergrößern und dann zu bearbeiten. Während des Bearbeitens lässt sich der Ausschnitt mit Hilfe der Pfeiltasten bewegen. Der Aufbau der vergrößerten Pixel kann – je nach Prozessor – einige Augenblicke dauern. Am Testrechner (486SX-25) war die Geschwindigkeit jedoch vollkommen ausreichend. Ob sich generell an der Geschwindigkeit noch etwas tut bleibt abzuwarten – nur damit keine Mißverständnisse aufkommen: Das Arbeiten geht ausreichend schnell von der Hand – auch bei den größten Punktstärken. An einem 80286-10 könnte es durchaus passieren, dass Zwangspausen eingelegt werden müssen, wenn beispielsweise die Füllfunktion genutzt wird.

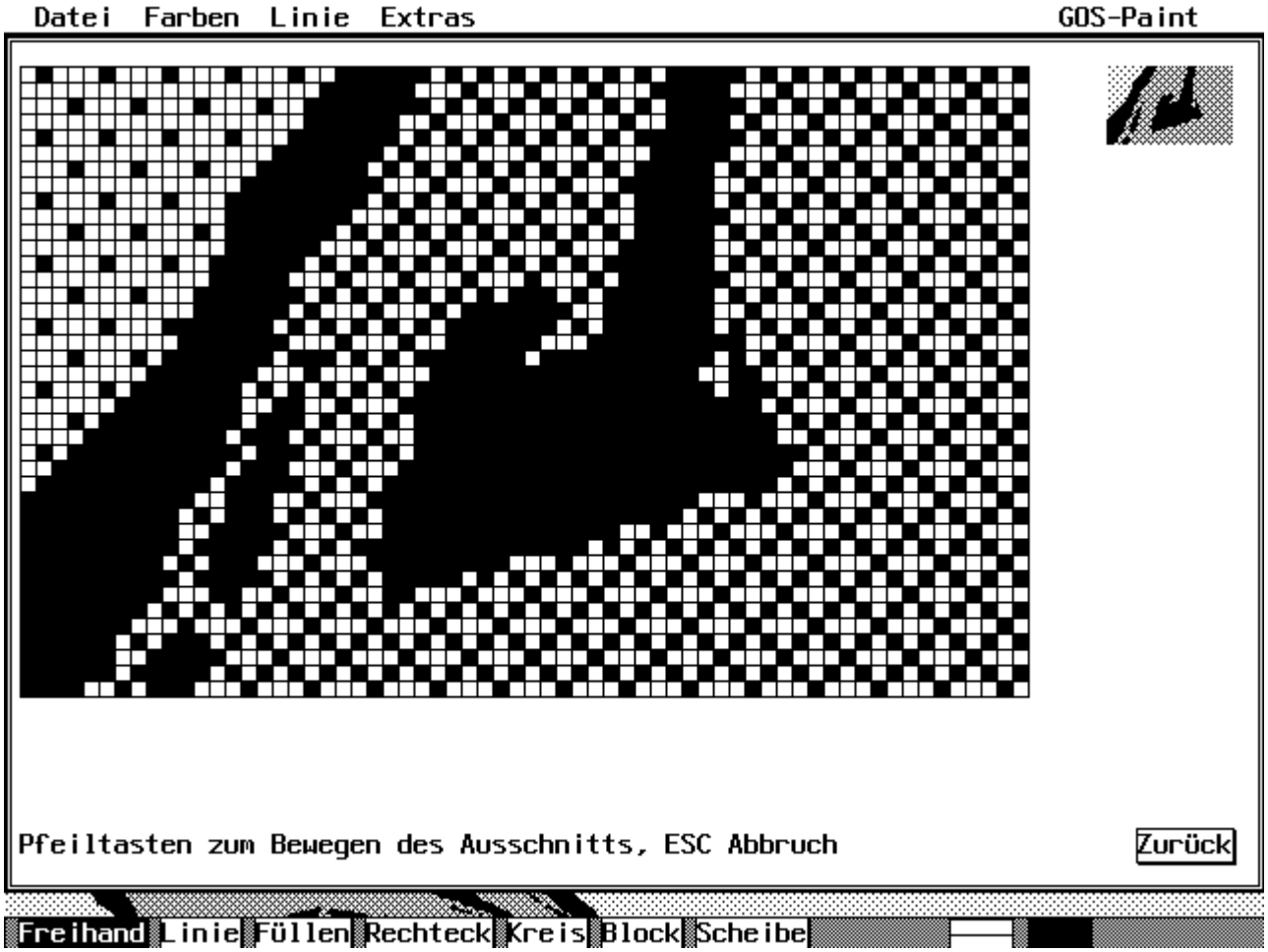
Sehr nützlich ist die Möglichkeit, beliebig große Bildausschnitte als BMP-Bild zu speichern. Weniger gut ist die fehlende „Stempel-Funktion“, um diese Ausschnitte per Maus in ein Bild einzufügen. Hoffentlich wird es hier noch einen Nachschlag geben – die User werden es sicher dankbar aufnehmen. Dafür wurde an eine wahlweise nutzbare Koordinatenanzeige gedacht. Dies erleichtert das exakte Malen ungemein. Wie oben bereits angedeutet, stellt GOS-Paint bis zu 8 beliebig definierbare Füllmuster zur Verfügung. Diese lassen sich über einen integrierten Editor erstellen und auch Speichern/Einladen. So kann sich der Anwender beliebige Muster-Bibliotheken zusammenstellen und bei Bedarf laden.

Etwas nervig empfindet man sicher die Füllfunktion. Sie ist auch der Grund, warum GOS-Paint 1 MB Arbeitsspeicher voraussetzt. Die Funktion FloodFill im GOS-eigenen Grafiktreiber muss, damit Füllmuster korrekt bearbeitet werden können, nämlich eine virtuelle Bitmap anlegen, um ein zweites Bit für jeden Pixel zu simulieren. Nur so ist gewährleistet, dass wirklich jedes Füllmuster angewendet werden kann. Leider benötigt diese zusätzliche BitMap Arbeitsspeicher, der unter GOS sowieso schon knapp ist. Deswegen lagert GOS klugerweise Programmteile in den erweiterten Speicher aus. Verfügt Ihr Rechner nur über 640K, fällt die Füllfunktion daher aus.

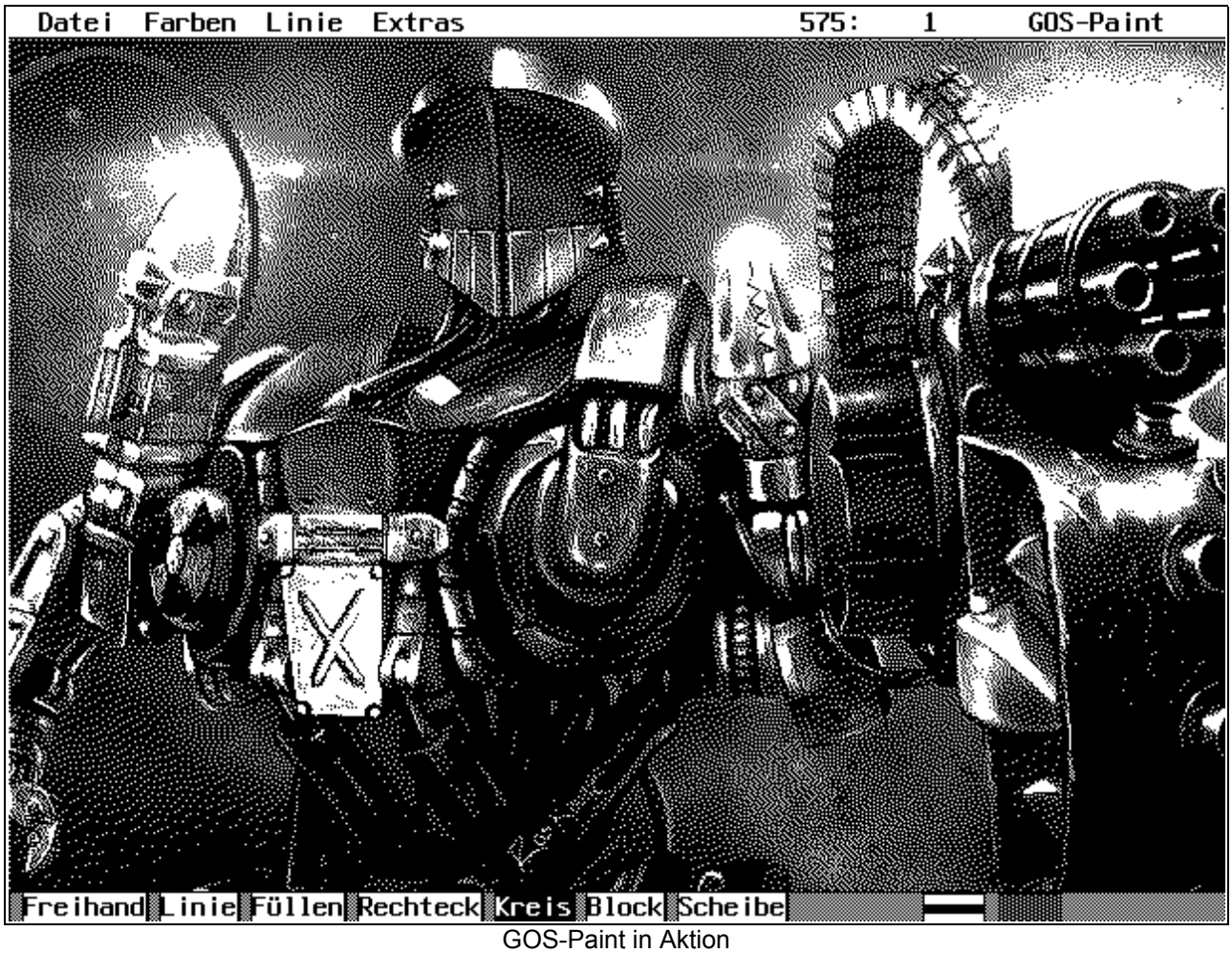
Zwar erlaubt GOS auch das Einstellen der Füllweite (diese bestimmt, wie genau Flächen gefüllt werden) – unter GOS-Paint ist von dieser Möglichkeit der Anpassung nichts zu finden. Deswegen kommt es bei schmalen Flächen durchaus vor, dass die Fülloperation plötzlich abbricht, obwohl noch nicht alle Punkte bearbeitet wurden. Wie bereits oben erwähnt, ist dies durchaus verbesserungswürdig!

Fazit

Trotz der beschriebenen Mängel ist GOS-Paint als brauchbares Programm zu bezeichnen – schon alleine deshalb, weil es das erste seiner Art ist. Beherrscht das Programm einmal eine Undo-Funktion und das Einfügen von „Stempeln“, geht erst so richtig die Post ab. Bis dahin heißt es aber, sich am Vorhandenen zu erfreuen.



Die Vergrößerung erlaubt das genaue Setzen/Löschen von Bildpunkten



GOS-Paint in Aktion

Eine neue TPU stellt sich vor

GOS286.TPU V2.0

Hinter den Kulissen brodelt es – eine neue Version der GOS286.TPU steht in den Startlöchern! Vergessen Sie alles, was Sie über Dialoge, Buttons, usw. wissen – die neue Version nimmt Ihnen alle Sorgen ab. GOS-Anwendungen werden nun spielend leicht erstellt.

Die neue Turbo Pascal Unit könnte wohl als Traum eines jeden Programmierers bezeichnet werden. Nahezu alle GOS- und Grafiktreiber-Funktionen sind über einfache Prozeduren nutzbar. Die Zeiten der Assembler-Freaks scheinen offenbar gezählt zu sein. Zum ersten Mal wurden Dialog-Funktionen eingeführt, die das Umständliche Neuanlegen von z. B. Menüs, bei denen sich ein Flag geändert hat, überflüssig macht.

Für alle GUI-Objekte können Flags und Inhalte nun direkt per Befehl beeinflusst werden. Damit zeigt sich Pascal nun endgültig einem Assembler ebenbürtig. Vergleichen wir einmal, die verschiedenen Möglichkeiten, einen Button (Nummer 1) hervorzuheben. Das Reverse-Flag befindet sich an Offset $000D der entsprechenden Struktur:

Assembler	GOS286.TPU 1.x	GOS286.TPU 2.0
`MOV BYTE [bstrukt+$0D],$FF`	`GETPOINTER($0301,s,o);` `MEM[s:o+$0D]:=$FF;`	`SetVFlag($0301,$FFFF);`

Ebenso kinderleicht lassen sich Inhalte von Textfeldern setzen oder auslesen – der Umweg über kompliziert anmutende FOR-MEM-INC-Konstrukte gehört der Vergangenheit an. Egal ob ViewPort, Mauszeiger oder Schriftarten – dem Programmierer stehen jetzt alle Wege offen. Als besonderes „Bonbon“ sei noch erwähnt, das die lange vermissten Grafikfunktionen (Fill, Circle, Line, usw.) ebenfalls enthalten sind. Eine erste Version der neue TPU ist auf der Homepage, sowie in der Yahoo!-Gruppe „bonnydos286“ erhältlich.

Als Fazit kann man sagen, dass mit dieser TPU wesentlich schneller und effektiver gearbeitet werden kann, als mit der vorherigen Version. Bis auf wenige Ausnahmen (Interrupt-Routinen) kann auf Assembler komplett verzichtet werden. Was jetzt noch fehlt, soll in Kürze noch nachgereicht werden: Ein besserer Zugriff auf erweiterten Speicher (ohne direkten Zugriff auf EXTMEM.APL), sowie ein verbesserter Systemdialog, der als Ersatz zur GOS-eigenen AlertBox dient.

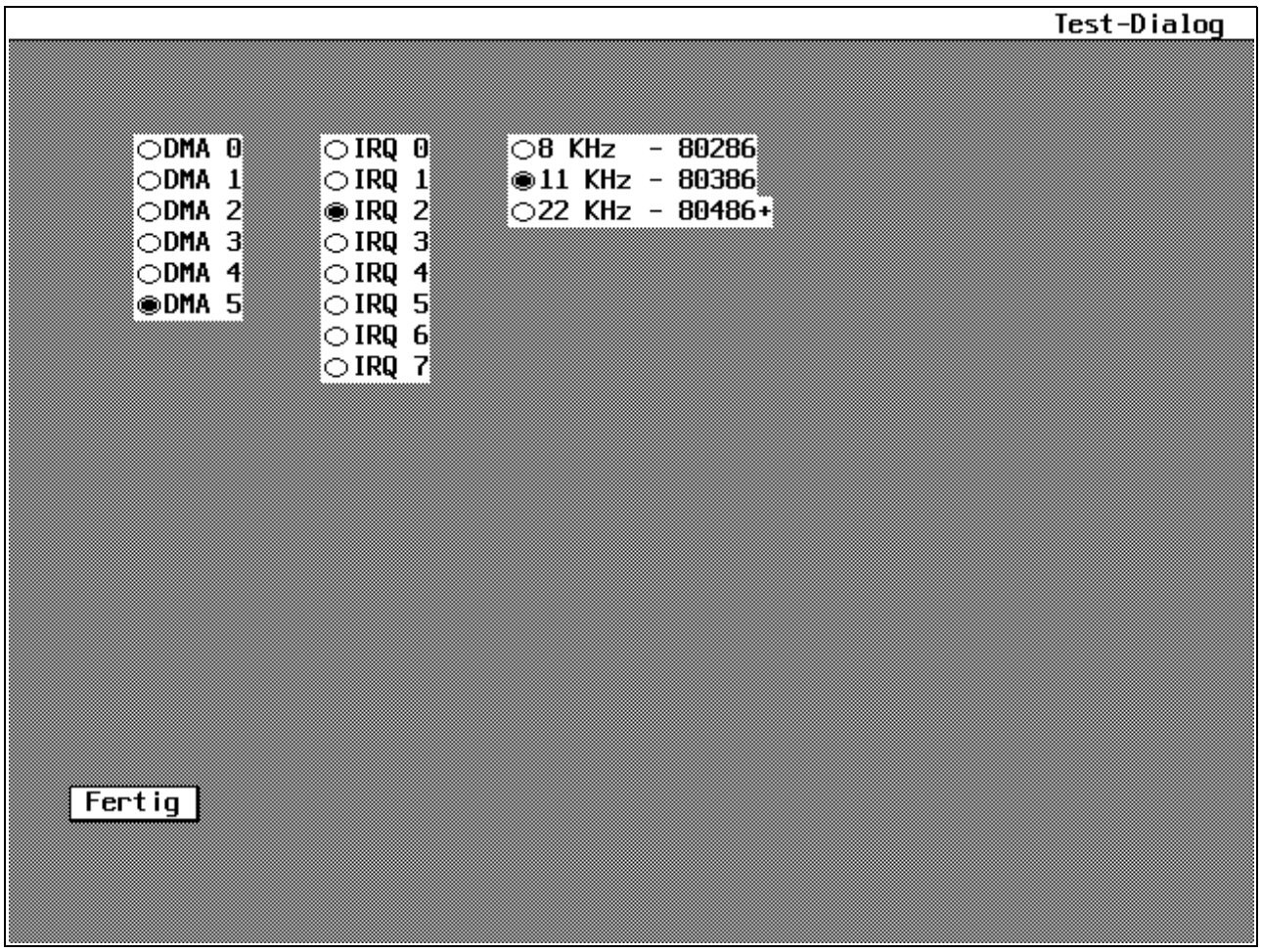
Eine neue GOS/286-Version stellt sich vor

GOS-Update verfügbar

Fast unbemerkt, und noch nicht in der Yahoo!-Group erhältlich, wurde ein aktuelles GOS/286-Archiv hochgeladen. Die wesentlichen Neuerungen beziehen sich dabei fast ausschließlich auf den Grafiktreiber GRAPHIC.DRV, sowie auf WritePad.

Im Wesentlichen wurden nun so genannte Option-Gruppen als mögliche GUI-Objekte hinzugefügt. Sie sollen die Programmierung von Dialogen stark vereinfachen und nebenbei ein besseres Aussehen haben, als die bisher verwendeten CheckBoxes (die natürlich immer noch enthalten sind). Um die Darstellung der Option-Gruppen zu beschleunigen, bestehen diese aus bestimmten ASCII-Zeichen. Insgesamt bleibt hier festzuhalten, dass Zeichen unterhalb CHR(32) nicht mehr zur Textausgabe genutzt werden sollten, da hier ggf. zukünftig weitere Dialog-Elemente hinterlegt werden.

Das erste „Opfer“ dieser Neuerung war WritePad – der Menüpunkt „Texthöhe“ wurde geändert, da hier durch den Doppelpfeil bereits optische „Nebenwirkungen“ der Option-Gruppen zu sehen waren. Durch das Update hat sich die Zahl der Funktionen im Grafiktreiber auf stolze 44 erhöht!



Die Option-Gruppen in einem Beispiel-Dialog zur Konfiguration einer Soundkarte

Erste Schritte mit Pascal – Teil 1

Wir basteln uns ein GOS-Programm

Die neue GOS286.TPU ist verfügbar – was spricht also dagegen, ein GOS-Programm zu schreiben? Dieser Kurs soll Ihnen die Programmierung von GOS/286-Programmen näher bringen und Sie mit den gängigen Strukturen vertraut machen.

In diesem Teil lernen wir die Grundlagen eines jeden GOS-Programms: Die Abfrage von Ereignissen, sowie das Erstellen und Manipulieren einer Schaltfläche (Button). Bevor wir anfangen, benötigen Sie noch etwas Handwerkszeug. Da wäre zum Einen Turbo Pascal 7.0, sowie den Inhalt des Archivs GOSTPU.ZIP. Die Unit GOS286.TPU kopieren Sie am Besten gleich in das Unit-Verzeichnis Ihrer Turbo Pascal-Installation.

Um aus der fertig kompilierten EXE-Datei eine BonnyDOS- bzw. GOS-Applikation zu machen, sollten Sie ebenfalls EXE2APL, sowie die beiden Dateien LOADEXE.BIN und LOADGOS.BIN bereit halten – kopieren Sie dies in das BIN-Verzeichnis von Pascal – dort befindet sich auch TURBO.EXE, sowie der Compiler TPC.EXE. Alles bereits? Dann geht's los! Wir beginnen mit einem neuen Programm (Menü „File"/"New").

Zuerst muss Pascal wissen, welche Units wir verwenden. Für unsere Zwecke reicht erstmal eine – die GOS286.TPU. Wir schreiben also als erstes:

```
USES GOS286;
```

Jetzt benötigen wir noch ein paar Variablen:

```
VAR

e1,e2,e3,e4:            Word;
vflag:                  Word;
p:                      Pointer;
```

Die ersten vier Wörter E1, E2, E3, und E4 benötigen wir für die Event-Abfrage. VFLAG wird unser Visible-Flag der Schaltfläche, der Zeiger P dient uns zur Ablage des Dialogs.

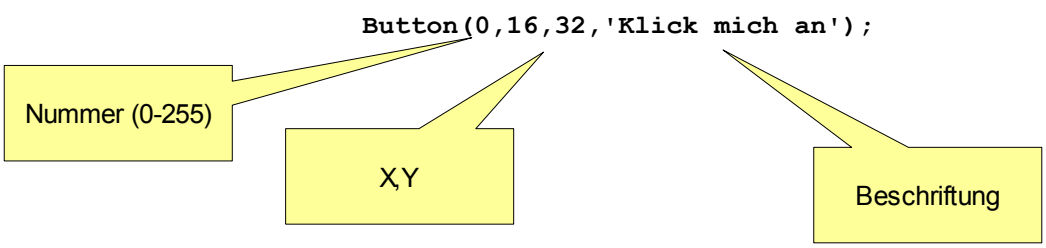
Was genau ist eigentlich ein Dialog? Ein Dialog ist die Grundlage aller GOS-Programme. Es ist eine verkettete Liste von Strukturen. Diese Strukturen beschreiben GUI-Objekte, wie eben unsere Schaltflächen. Beachten Sie, dass immer(!) eine Struktur im Dialog enthalten sein muss – auch wenn Sie gar keine GUI-Objekte verwenden. Selbst bei der Abfrage der Tastatur muss diese Regel stets beachtet werden! Definieren Sie im Zweifelsfalle einfach einfach eine Schaltfläche außerhalb des sichtbaren Bildschirms (z. B. an Position $A000,$A000).

Wie legt man einen Dialog an? Dazu reservieren wir im ersten Schritt etwas Speicher:

```
begin
 GetMem(p,256);
 UseDialog(p);
```

Wir benötigen ja nur Platz für einen Button – deshalb reichen uns 256 Bytes. Legen Sie mehr Schaltflächen an, erhöhen Sie den Wert entsprechend. Beachten Sie, dass weder Turbo Pascal, noch die GOS286.TPU prüft, ob genügend Speicher für einen Dialog reserviert wurde. Dies obliegt Ihrer Verantwortung. Der Befehl `USEDIALOG` teilt der GOS286.TPU mit, welcher Speicherbereich nun für den aktuellen Dialog genutzt wird.

Die Schaltfläche ist schnell erstellt:



Zwar beinhaltet unser Dialog jetzt einen Knopf – doch auf dem Bildschirm ist noch nichts zu sehen. Beachten Sie also: Jeder Dialog muss gezeichnet werden! Dies übernimmt für uns GOS/286:

```
 ReDrawGUI(p);
```

Unser Programm soll nun warten, bis die Schaltfläche angeklickt wurde, und sich dann beenden:

```
Repeat
 GetEvent(p,e1,e2,e3,e4);
Until e1=ButtonSelect;

end.
```

Da wir nur eine einzige Schaltfläche (mit der Nummer 0) angelegt haben, reicht es aus, die Ereignisklasse (E1) zu testen. Unsere Repeat-Until-Schleife läuft also solange, bis die Ereignisklasse BUTTONSELECT eintritt. Werfen Sie an dieser Stelle einmal einen Blick in HEADER.TXT (im GOS286.TPU-Archiv) – hier sehen Sie, welche Ereignisse welchen numerischen Wert besitzen.

Mögliche Erweiterungen

Selbstredend, dass eine Schaltfläche recht langweilig ist. Von jeder Objektklasse kann GOS max. 256 Objekte verwalten. Das heißt, wir können für weitere Schaltflächen Nummern von 0 bis 255 vergeben, wobei Nummer 0 bereits verwendet wird. Es spricht zwar nichts gegen die mehrfache Vergabe von identischen Nummern – die genaue Zuordnung des Objekts ist dann aber nicht mehr möglich.

Wir schreiben uns also folgendes Programm, das wir im BIN-Verzeichnis von Pascal unter `DEMO.PAS` speichern:

```
uses gos286;

var

e1,e2,e3,e4:            Word;
vflag:                  Word;
p,o:                    Pointer;

label schleife;

begin

 GetMem(p,256);
 UseDialog(p);

 Button(0,16,32,'Klick mich an');
 Button(1,16,50,' Beenden ');

 RedrawGUI(p);

schleife:
Repeat
 GetEvent(p,e1,e2,e3,e4);
Until e1=ButtonSelect;

if e2=1 then Exit;      { Beenden bei Nr. 1 }

GetVFlag($0300,vflag);
vflag:=vflag xor $FF00;
SetVFlag($0300,vflag);

GetPointer($0300,o);    { Zeiger auf Struktur des Buttons 0 holen }
ReDrawSingle(o);  { Neu zeichnen von Button 0 }
goto schleife;
end.
```

Jetzt sieht unser Programm schon wesentlich komplexer aus. Was genau tut dieses aber? Es legt zwei Schaltflächen an und fragt diese per `GetEvent` ab. Wird Schaltfläche Nummer 1 betätigt, wird das Programm beendet. Einen Test auf Nummer 0 können wir uns sparen – da wir nur zwei Buttons haben. Beim Klick auf Schaltfläche 0 passiert folgendes:

Es wird das Visible-Flag des Buttons 0 geholt. Schaltflächen nutzen dieses Flag wie folgt:

Hi-Byte	Lo-Byte
0: Normale Darstellung <>0: Reverse Darstellung	0: Visible = False <>0: Visible = True

Mit `GetVFlag` holen wir uns das Visible-Flag der Schaltfläche 0, kippen alle Bits im Hi-Byte und legen dieses Flag mit `SetVFlag` wieder ab. Anschließend holen wir uns den Zeiger auf die Button-Struktur und lassen das Objekt mit `ReDrawSingle` neu zeichnen – ansonsten würden wir die Änderungen nicht sehen.

Experimentieren Sie ruhig einmal – setzen Sie doch einmal das Lo-Byte des Visible-Flags auf $00. Nach dem Neuzeichnen ist die Schaltfläche nun gesperrt – wird also in Geisterschrift dargestellt und kann nicht mehr angeklickt werden. Als letztes Rätsel bleibt noch der seltsame Parameter $0300 bei `Get/SetVFlag` und `GetPointer`. Auch dieses Rätsel ist schnell gelöst. Doch zuerst schauen wir uns noch die Struktur einer Schaltfläche an, so wie sie durch `BUTTON` im Dialog-Speicher angelegt wird:

Offset	Inhalt
`$0000`	Segment nächste Struktur
`$0002`	Offset nächste Struktur
`$0004`	Hi-Byte: Objektklasse $03 Lo-Byte: Button-Nummer
`$0006`	X
`$0008`	Y
`$000A`	Breite (wird von GOS/286 eingetragen)
`$000C`	Hi-Byte: $00 normal, $FF Reverse Lo-Byte: $00 visible false, $FF visible true
`$000E`	Beschriftung mit abschließendem Null-Byte

Wenn Sie sich nun einmal das Wort an Offset $0004 ansehen, dann kommen Sie sicher auf die Lösung des oben genannten Rätsels. `SetVFlag($0300,....)` heißt also, es soll das VisibleFlag (Offset $000C) der Schaltfläche ($03) Nummer 0 ($00) – kurz Objekt $0300 – gesetzt werden. Das gleiche Schema nutzt auch `GetPointer` und wird Ihnen beim Umgang mit Dialogen des Öfteren begegnen.

Doch Vorsicht: Als Grundlage zur Suche nach einem Objekt wird immer der mit `UseDialog` gesetzte Speicherbereich genutzt!

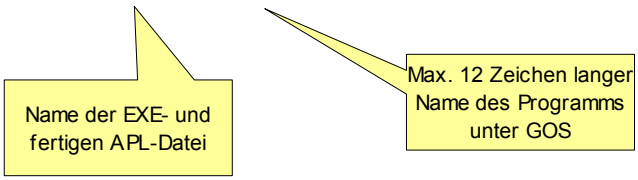
Wir kompilieren unser Programm

Speichern Sie das eben erstellte Programm ab. Wechseln Sie nun in die Eingabeaufforderung und geben Sie ein:

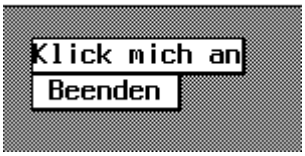
```
TPC DEMO.PAS
```

Kein Fehler? Prima! Nun haben wir zwar eine DEMO.EXE – die nützt uns aber nicht viel. Wie anfänglich vorgeschlagen, sollte sich noch EXE2APL im BIN-Verzeichnis von Turbo Pascal befinden:

```
EXE2APL DEMO Demo-APL
```



Kopieren Sie DEMO.APL nun auf eine Diskette und führen Sie das Programm unter GOS/286 aus. Die beiden folgenden Screenshots zeigen, wie sich die Schaltfläche Nummer 0 („Klick mich an“) verhält, wenn Sie diese anklicken. Die Darstellung wechselt bei jedem Klick zwischen Normal und Reverse.



Was wäre wenn...

Was wäre passiert, hätten wir zum Neuzeichnen der Schatlfäche 0 nicht `ReDrawSingle`, sondern wieder `ReDrawGUI` geschrieben? Nun, in unserem Programm wäre genau das gleiche passiert. Bei komplexeren Dialogen ist `ReDrawGUI` jedoch langsamer – es zeichnet nämlich ALLE Objekte, die ab dem anzugebenden Pointer angelegt wurden. Gewöhnen Sie sich daher an, einzelne Objekte immer mit `ReDrawSingle` zu zeichnen, wenn diese geändert werden.

Im Teil 2 lernen wir weitere GUI-Objekte kennen und steigen noch tiefer in die GOS-Programmierung ein. Bis dahin: Viel Spaß beim Experimentieren!