ShellScr_german

Kyzer/CSG

COLLABORATORS

	TITLE : ShellScr_german	

ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Kyzer/CSG	April 15, 2022	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

ShellScr_german

1.1 ShellScr v1.6 Anleitung

ShellScr v1.6

Mit ShellScr lässt sich eine Shell in vollen Ausmaßen auf einem eigenen Screen öffnen.

```
  Free Software Foundation, Inc.
  59 Temple Place - Suite 330
  Boston, MA 02111-1307
  USA
```

Wie erreiche ich den Autor?

1.2 Einführung

Warum sollte man die Shell nicht auf ihrem eigenen Screen öffnen, ↩
anstatt sie in einem Fenster laufen zu lassen und damit die Workbench unnötig zu füllen?

ShellScr ist ein Programm, daß einen neuen öffentlichen Screen öffnet und darauf eine Shell in den vollen Ausmaßen des Screens öffnet. Nachdem alle Fenster auf diesem Screen geschlossen wurden (inklusive der Shell), schließt er sich automatisch.

ShellScr kann die Shell auch mit einem anderen Skript als s:shell-startup starten und bietet sich somit als Alternative zu IconX an.

Wie ShellScr benutzt wird.

1.3 ShellScr benutzen

Systemvoraussetzungen:

ShellScr benötigt mindestens Workbench 2 (v37). Um einen anderen Zeichensatz als Topaz benutzen zu können, muss die diskfont.library installiert sein.

Optional können ASL-Requester zum Auswählen des Zeichensatzes und des Screens genutzt werden. Dafür wird mindestens Workbench 2.1 (v38) benötigt oder es muß ein Ersatz wie ReqTools oder ReqPath vorhanden sein.

Verteilung:

Bitte gib nur das ganze Archiv weiter, so wie du es bekommen hast, inklusive Source, Beispielicons und dieser Anleitung. Danke.

Installation:

Einfach das Installationsprogramm starten.

ShellScr starten:

Du kannst soviele Kopien von ShellScr starten, wie du möchtest. ShellScr kann auch resident gehalten werden.

Um ShellScr zu starten, klicke einfach zweimal auf das Icon, oder gib "ShellScr" in eine bereits geöffnete Shell ein. Damit öffnet sich eine neue Shell auf ihrem eigenen Screen.
Dieser Screen ist jetzt der Standard-Screen, viele Programm werden ihre Fenster ebenfalls auf ihm statt auf der Workbench öffnen.

Wenn du eine Schublade oder eine Datei auf der Workbench auswählst und dann mit Shift ShellScr startest, wird die neue Shell als erstes diese Schublade bzw die Schublade, in der die ausgewählte Datei liegt, zum Arbeitsverzeichnis erklären.
Dies funktioniert auch, wenn ShellScr mit Hilfe von Programmen wie "ToolsDaemon" oder "ToolManager" gestartet wird.

Die Shell kann jederzeit wieder geschlossen werden, indem entweder "EndShell" eingeben wird oder man CTRL-\ drückt.
Der Screen wird sich allerdings erst schließen, wenn alle anderen Fenster, die auf ihm noch offen sind, geschlossen wurden. Normalerweise ist die Shell jedoch das einzige Fenster, so daß sich der Screen schließt, wenn die Shell geschlossen wird.

Screen und Shell können mit
Optionen
angepasst werden.

1.4 ShellScr-Optionen

Mit den folgenden Optionen kannst du bestimmen, wie Screen ↩
und Shell
arbeiten.

Du kannst sie beim Aufruf in der Shell angeben, als Tooltypes im ShellScr-Icon, als Tooltypes von einem Icon, daß ShellScr als Standardprogramm hat oder als Tooltypes von einer Schublade oder Datei, die ausgewählt sind, wenn ShellScr gestartet wird.

Optionen für den Screen

PUBNAME

MODEID

DEPTH

FONT

AUTOSCROLL

```
                SHANGHAI


                TITLE


                NOTITLE

Optionen für die Shell


                CONSPEC      (WINDOW)


                COMMANDFILE  (FROM)


                STACKSIZE
```

1.5 PUBNAME Option

```
                WB:     PUBNAME
Shell:  PUBNAME=NAME
```

Dies ist der Name, den der von ShellScr geöffnete, öffentliche Screen erhält.

Standardmäßig wird SHELL_XXXX benutzt, wobei XXXX eine eindeutige Zahl für diese Shell ist.

Du solltest diese Option nur benutzen, wenn es unbedingt nötig ist, da es nicht möglich ist, zwei Screens mit demselben Namen zu öffnen.
Es besteht die Möglichkeit, mit MultiCX dem vordersten Screen, egal welcher es ist, einen speziellen Namen zu geben.

Um dem Screen einen bestimmten Titel zu geben, kannst du

```
                TITLE
```

benutzen.

1.6 MODEID Option

```
                WB:     MODEID
Shell:  MODEID=ID
```

Dies ist die ModeID des Modus (dh die Auflösung), den der Screen haben soll.

Es gibt folgende Möglichkeiten, den gewünschten Modus anzugeben:

- als Dezimalzahl (kompatibel zu den ModeIDs in älteren ShellScr-Versionen).
- als Hexadezimalzahl, beginnend mit einem '$' oder '0x', zB "0x29004"
- mit dem Namen des Modus, zB "PAL:High Res Laced"
- mit einem Fragezeichen, "?", oder einem leeren Argument, "". In diesem Fall wird ein Requester geöffnet, in dem der Modus eingestellt werden kann.

Eine Liste aller verfügbaren Modusnamen findest du im Screenmode-Voreinsteller oder in dem Screenmode-Requester, den ShellScr öffnet (s.o.).
Beim Vergleich der Modusnamen wird zwischen großen und kleinen Buchstaben unterschieden; "PAL:High Res" ist also nicht das gleiche wie "pal:high res".

Wenn kein Modus angegeben wird oder der angegebene Modus nicht benutzt werden kann, wird der Modus des aktuellen Standardscreens (normalerweise die Workbench) benutzt.

Siehe auch: die
DEPTH-Option
zum Bestimmen der Farbanzahl des Screens.

1.7 DEPTH Option

WB: DEPTH
Shell: DEPTH/N

Dies ist die Anzahl Bitplanes, die der neue Screen haben soll. Diese Zahl, die "Tiefe", bestimmt, wieviele Farben auf einem Screen maximal dargestellt werden können.
Auf normalen Amigas kann sie zwischen 1 und 4 liegen, auf AGA-Maschinen sind bis zu 8 möglich, mit Grafikkarten bis zu 16 (Hicolor) oder auch 24 (Truecolor).

Die Anzahl Farben ergibt sich aus der Rechnung 2^n, wobei n die Anzahl Bitplanes ist (zB 2^8=256 Farben).
Je mehr Bitplanes benutzt werden, desto mehr Speicher benötigt der Screen.

Standard sind 2 Planes (4 Farben). Für Benutzer der MagicWB sind vielleicht 3 angebracht, um alle 8 MWB-Farben darstellen zu können.

In dem Screenmode-Requester, der sich mit der
MODEID-Option
öffnen lässt,
kann die Farbtiefe ebenfalls eingestellt werden.

Die Farben des neuen Screens werden vom Standardscreen kopiert.

1.8 FONT Option

```
WB:     FONT
Shell:  FONT/K
```

Mit dieser Option lässt sich ein Zeichensatz (mit fester Breite) für den Screen einstellen. Gib einfach den Namen und die Größe an, zB "XEN/9". Falls der Zeichensatz die standardmäßigen 8 Punkt groß sein soll, kann die Größenangabe auch weggelassen werden.

Mit "" oder "?" wird ein ASL-Requester geöffnet, in dem der gewünschte Zeichensatz eingestellt werden kann.

Shells übernehmen normalerweise nur Zeichensätze mit fester Breite. Es gibt keinen Grund, dich daran zu hindern, einen proportionalen Zeichensatz zu verlangen; ShellScr wird aber eine Warnung ausgeben und möglicherweise wird die Shell einen anderen benutzen.

Wird FONT weggelassen, sucht sich ShellScr den passenden Zeichensatz in folgender Reihenfolge: Wenn der Standardscreen einen Zeichensatz mit fester Breite benutzt, wird ShellScr diesen nehmen, anderfalls wird der Zeichensatz benutzt, der als Systemzeichensatz im Font-Einsteller definiert wurde.

1.9 AUTOSCROLL Option

```
WB:     AUTOSCROLL
Shell:  AUTOSCROLL/S
```

Ist diese Option angegeben, wird der Autoscrollmodus des Screens aktiviert. Falls der Screen vom Bildschirm gezogen wurde, kann er jetzt mit einer einfachen Mausbewegung zurückgeschoben werden.

1.10 SHANGHAI Option

```
WB:     SHANGHAI
Shell:  SHANGHAI/S
```

In den Versionen vor AmigaOS 2 hatte ein Programm nur zwei Möglichkeiten ein Fenster zu öffnen: entweder auf einem eigenen Screen oder auf der Workbench.

Seit AmigaOS 2 werden Fenster normalerweise auf dem Standardscreen geöffnet. Um kompatibel zu den älteren Versionen zu bleiben, erscheinen die Fenster von alten Programmen weiterhin auf der Workbench.

Es gibt jedoch eine spezielle Option für öffentliche Screens namens "Shanghai". Ist diese für einen öffentlichen Screen eingeschaltet, werden auch Fenster von alten Programmen auf diesem Screen geöffnet.

1.11 TITLE Option

```
                WB:     TITLE
Shell:  SCREENTITLE=TITLE
```

Dies ist der Text, der im Titel des Screens erscheint. Diese Option ist eingebaut, weil das Shellfenster selber keinen eigenen Titelbalken hat. Die Vorgabe ist 'AmigaShell'.

Damit garkeine Titelzeile erscheint, muß
NOTITLE
gesetzt werden.

1.12 NOTITLE Option

```
WB:     NOTITLE
Shell:  NOTITLE=HIDETITLE/S
```

Ist diese Option angegeben, bekommt der Screen keinen Titelbalken. Damit hast du wirklich den kompletten Bildschirm für die Shell zur Verfügung.
Beachte, daß das normale CON:-Fenster mit dieser Option ziemlich unschön aussieht! Es wirkt besser bei einem Handler wie VNC:, der wirklich den ganzen Schirm nutzt.

1.13 CONSPEC (WINDOW) Option

```
                WB:     CONSPEC
Shell:  CONSPEC=WINDOW
```

Dies ist die technischste Option von ShellScr. Du brauchst sie nicht benutzen, wenn du nicht möchtest!

CONSPEC definiert das Fenster, daß auf dem neuen öffentlichen Screen geöffnet werden soll. Die Definition sollte zwei "%s" enthalten, die von ShellScr mit besonderen Werten ersetzt werden. Das erste steht für die Position und Größe des Fensters (im Format "l/r/b/h", zB "0/3/640/253" auf einem 640x256 großen Screen). Für das zweite "%s" wird der Name des Screens eingesetzt, auf dem das Fenster geöffnet werden soll. Das Fenster sollte ein Hintergrundfenster ohne Rahmen, Gadgets und Titelbalken sein.
Die Voreinstellung lautet

```
CON:%s//BACKDROP/NOBORDER/SCREEN %s
```

Falls du ViNCEd benutzt, ist

```
VNC:%s//SCREEN%s/BACKDROP/NOBORDER/NOCLOSE/NOSIZE/NODRAG/NODEPTH/NOPROPX/
NOPROPY/NOBUTTONS/NOICONIFY/SHELL/MENU
```

eine gute Wahl.

Du solltest auch einen Blick auf die
Berechnung der Fenstergröße
werfen.

1.14 COMMANDFILE (FROM) Option

```
WB:     COMMANDFILE
Shell:  COMMANDFILE=FROM
```

Mit dieser Option kannst du ein Script angeben, das beim Start von ShellScr ablaufen soll. Normalerweise ruft die Shell s:shell-startup auf. COMMANDFILE funktioniert genauso wie die FROM-Option von NewShell, genauer gesagt wird hier eben diese Option von NewShell benutzt. Abgesehen von ein paar Kleinigkeiten macht das auch IconX.

1.15 STACKSIZE Option

```
WB:     STACKSIZE
Shell:  STACKSIZE=STACK/N
```

Normalerweise übernimmt die neue Shell die Größe ihres Stapelspeichers (Stack) von ShellScr. Muß man also ShellScr einen großen Stapelspeicher geben (obwohl ShellScr nur 1k benötigt), damit auch die Shell einen bekommt? Natürlich nicht, mit dieser Option kann die Größe des Stapels für die Shell angegeben werden ohne den von ShellScr zu beeinflussen.

Die Größe wird in Bytes angegeben; sie wird von ShellScr auf die nächste 4-Byte-Grenze aufgerundet. Das Minimum sind 1600 Bytes, das Maximum ist nur vom verfügbaren Speicher begrenzt. Die Standardgröße ist 4096 Bytes.

1.16 ShellScr Geschichte

```
                1.0:    First release

1.1:    Added options
                PUBNAME
                ,
                MODEID
                ,
                DEPTH
                ,
                TITLE
                 and
                CONSPEC
                .
        Use System() instead of Execute().
        Tidied up source a bit.

1.2:    Made a few more checks on stuff.
        Added
                NOTITLE
                 option.
        Better
                windowsize calculation
                        Now sets currentdir and paths properly from Workbench.
```

```
1.3:    Now parses icon tooltypes.
        Now uses default screen's colours.
        Tidied up documentation.

1.4:    Screenmode and font requesters.
        Added
                FONT
                 option.
        Added
                COMMANDFILE
                 option.
        Fixed default conspec to act properly.
        Now explains errors a lot better, especially "can't open screen".
        Fixed args parsing. Basically, directories and iconless icons
        caused crashes/lockups/trashed mem/etc... not anymore.
        You can now make ShellScr resident.
        Popup 'Screen Closing' requester if ShellScr can't close its screen
        immediately.

1.4b:   Attempted bugfixes.
        Certainly fixed args.m from causing Enforcer hits.
        Removed the false calls if asl.library _didn't_ open.
        Fixed screen and font parsing a little.

1.5:    Added
                STACKSIZE
                ,
                AUTOSCROLL
                 and
                SHANGHAI
                 options.
        Set depth/autoscroll from the screenmode requester.
        Cancelling the screenmode requester now aborts running the program.
        'Screen Closing' requester now has 'Cancel' option.
        Now sets itself as the default public screen on opening.
        Added 'real name' screenmode recognition.
        Removed the 'hangup' that occurs the NewShell command fails.
        Error message when NewShell command fails.
        Fixed bugs with font selection.
        Numerous tweaks, rehashes and things I have forgotton.

1.6:    Fixed the
                well-reported bug
                .
        Added localization.
        Added happy Installer script.
        Font can be specified in 'myfont.24' format.
        Removed 'Screen closing' requester, as it proved very difficult to
        bring it up. The user can now send ShellScr a CTRL-C to reattempt
        closure, if the Intuition mechanism fails.
        Now asl.library is only opened if really neccessary.


                 Geplant..
```

1.17 Aussichten für ShellScr

Neue Ideen sind mir inzwischen ausgegangen.

Wenn du wirklich willst, kann ich Teile rausnehmen, die du nicht magst oder brauchst. Das alte scrsh war nur wenig größer als 1k (allerdings auch nicht ganz fehlerfrei und systemkonform -Ed.)

Vorschläge für neue Dinge sind natürlich auch willkommen -
schreib mir.

1.18 Anmerkungen zum Source

Das makefile ist nur für meine Maschine geschrieben und wird ShellScr nicht erzeugen. Weil aber alle Module vorhanden sind, sollte das kein Problem darstellen; schreib einfach "ec opti ShellScr.e".

Alle Module und ihre Quelltexte befinden sich in einem Archiv im Aminet unter "dev/e/kyz.lha". Solange du mich darüber informierst, kannst du sie alle, modifiziert oder nicht, weiterbenutzen.

Die Assembler-Module nutzen E globals mit Hilfe eines Includes, daß von eglobs.e erzeugt wurde.

Ich gebe keine von den Strings und Listen selbst zurück, sondern überlasse das dem Startup-Code. Das macht das Programm etwas kleiner und dem Quelltext übersichtlicher, was das bisschen mehr Speicher, den ShellScr dadurch beim Laufen braucht, wett macht.

Der große böse Fehler:

Es war ein Fehler in der clr()-Routine, der das erste Byte einer Addresse nach der args-Struktur gelöscht hat. Das konnte ich auf meinem 24bittigen EC020 natürlich nicht bemerken, aber jeder andere. Soll nicht wieder vorkommen :)

1.19 Berechnung der Fenstergröße

Die Fenstergröße wird folgendermaßen berechnet:

Die Breite des Fensters ist immer die gesamte Breite des Screens.

Die Höhe ist, sofern du nichts anderes per
CONSPEC
definiert hast, die Höhe
des Screens minus 3, weil ein CON:-Fenster mit dieser Einstellung am Besten aussieht; die Titelzeile wird vom Titel des Screens verdeckt.

Wenn du ein eigenes Fenster mit
CONSPEC
definiert hast, zieht ShellScr von

der Höhe des Screens die Höhe der Titelzeile ab. Hast du außerdem die Option

NOTITLE

angeben, wird gar keine Titelzeile angezeigt und du bekommst die volle Höhe des Screens.

1.20 Glossar wichtiger Ausdrücke

Screen

Ein Grafikbereich, der auf dem Monitor angezeigt wird.

Screens können (normalerweise) auf auf dem Monitor rauf- und runtergezogen werden und generell übereinander angeordnet. Die Anzahl der Screens ist nur durch den verfügbaren Speicher begrenzt.

Workbench Screen

Dies ist praktisch der Hauptscreen des Amigas. Er wird nach dem Starten des Rechners als erstes geöffnet.
Die Workbench ist ein öffentlicher Screen und wird als Standardscreen registriert, auf dem die Programme, die keine eigenen Screens öffnen, ihre Fenster öffnen (Standardscreen, su).

Custom Screen

Dies ist ein Screen, der von einem Programm nur für sich selbst geöffnet wird. Kein anderes Programm darf hier ein Fenster öffnen.

Öffentlicher Screen

Auch Public Screen. Dies ist ein Screen, auf dem jedes Programm ein Fenster öffnen kann. Jeder öffentliche Screen hat einen eindeutigen Namen (der der Workbench ist "Workbench"), den andere Programme beim Öffnen eines Fensters angeben können, um das Fenster auf diesem Screen erscheinen zu lassen.
Wird kein Name angegeben, öffnet sich das Fenster auf dem Standardscreen.

Standardscreen

Dies ist normalerweise die Workbench. Diese Vorgabe kann aber auch von Programmen, oder vom Nutzer mit entsprechender Software, geändert werden. Jeder öffentliche Screen kann Standardscreen werden.

ShellScr übernimmt die Vorgaben für viele Einstellungen vom Standardsceen. Wenn zB für

ID

nichts angegeben ist, wird ShellScr einen gleichen Screen wie den der Workbench öffnen

Der Standardscreen wird benutzt, um Fenstern, für die kein anderer Screen angeben wurde, eine "Heimat" zu geben. Es gibt jedoch eine Ausnahme - aus Kompatibilitätsgründen öffnen alte Programme ihre Fenster nach wie vor auf der Workbench. Dies kann jedoch mit dem

Shanghai-Modus

geändert werden.

Stapelspeicher (Stack)

Ein Speicherbereich, den jedes Programm benötigt um Daten für kurze Zeit zwischenzuspeichern.
UNIX-Programme sind bekannt für ihren enormen Stackverbrauch.

Source

Der Quelltext, den ein Programmierer schreibt und anschließend in ein ausführbares Programm übersetzt.

1.21 Credits

ShellScr ist aus dem Assembler-Programm 'scrsh' von Kyzer/CSG entstanden.

ShellScr wurde von Kyzer/CSG in Amiga E 3.3a geschrieben, basierend auf dem Sourcecode von scrsh.asm von Kyzer/CSG, dem RKM- Sourcecode für clonescreen.c, und dem Sourcecode für dospath.library von Stefan Becker. Das Programm endhält jedoch keine Teile mehr von diesen Quellen.

Die Übersetzungen wurden von der Amiga Translators Organisation gemacht.

Grüße an die folgenden Personen:
(DE) Walter Haidinger und deJoker, die beide nach der COMMANDFILE-Option gefragt haben.
(AU) Frank Bunton, der sich besser Farben wünschte.
(DE) Thomas Richter, für die "Reparatur" von WBLoad, und für ViNCEd!
(NL) Wouter van Oortmerssen, für die Sprache E.
(DE) Frank Wille, für PhxAss.
(US) Free Software Foundation, für die GNU Lizenz und GNU make.

Copyrights:
IconX: © Amiga International (so steht es hier...)
MultiCX: ©~Martin Berndt.
PubChange: © Steve Koren.
ReqTools und ReqPatch: © Nico François, Magnus Holmgren und Dave Jones.
ToolManager: © Stefan Becker
ToolsDaemon: © Nico François
ViNCEd und VNC: © Thomas Richter.
XEN: © Martin Huttenloher

Kontakt:

Kyzer/CSG,
49 Fairview Road,
AB22 8ZG, Scotland.

oder per ePost: kyzer@4u.net

http://zap.to/kyz