Fortsetzung von Ausgabe 7/93, Seite 64

Die Systembefehle (Folge 6)

Referenz: Shell 2.1/3.0

von Peter Aurich

Reboot -> Neustart

reentrant -> Programm, wiedereintrittsfähig

RELABEL Shell-Befehl bzw. Systemprogramm; ändert den Namen der Diskette im angegebenen Laufwerk. *Befehlsformat: RELABEL [DRIVE] <Laufwerk>: [NAME] <Name>* Beispiele:

```
relabel df1: DinglDongl
relabel Empty Full
```

REMRAD Shell-Befehl bzw. Systemprogramm; entfernt die resetfeste RAM-Disk -> RAD (Parameter FORCE) bzw. löscht alle Dateien darin und verringert die Größe auf ein Minimum. *Befehlsformat: REMRAD [<Laufwerk>] [FORCE]*

Die Angabe des Laufwerks ist nur erforderlich, wenn mehrere resetfeste RAM-Disks angemeldet sind. Beispiel:

```
remrad force
```

RENAME Shell-Befehl bzw. Systemprogramm; ändert den Namen einer Datei oder eines Verzeichnisses. *Befehlsformat: RENAME [FROM] {<NameAlt>} [TO|AS] <NameNeu> [QUIET]*

Bei Angabe unterschiedlicher Pfade beim alten und neuen Namen wird die Datei oder das Verzeichnis verlegt. Mit QUIET verzichtet RENAME auf eine Bestätigung am Bildschirm. Beispiele:

```
rename bilder/pic bilder/Bild
rename df0:brief1 df0:brief2 to
dh0:briefe
```

REQUESTCHOICE -> Shell-Befehl bzw. Systemprogramm; zeigt eine definierbare Dialogtafel mit Meldung und Symbolschaltern am Bildschirm, wartet auf einen Mausklick und speichert die Nummer des angewählten Symbolschalters in der Umgebungsvariablen -> Result2. *Befehlsformat: REQUESTCHOICE Titel Text Symbole [PUBSCREEN <Public-Screen>]*

Enthält ein Argument Leerzeichen, muß es in Anführung stehen. Die Texte für die Symbolschalter sind mit »|« zu trennen. Beispiel:

```
requestchoice "Bilder anzeigen "
Weitermachen? Ja|Nein
```

REQUESTFILE -> Shell-Befehl bzw. Systemprogramm; bringt ein Dateiauswahl-Dialogfenster variabler Voreinstellung auf den Bildschirm, wartet auf die Auswahl eines oder mehrerer Einträge, gibt die entsprechenden Namen im Shell-Fenster aus und liefert die Rückgabe 0 (keine Auswahl) oder 5. *Befehlsformat: REQUESTFILE [DRAWER <Schublade>] [FILE <Datei>] [PATTERN <Namensmuster>] [TITLE <Titel>] [POSITIVE <Text>] [NEGATIVE <Text>] [ACCEPTPATTERN <Namensmuster>] [REJECTPATTERN <Namensmuster>] [SAVEMODE] [MULTISELECT] [DRAWERSONLY] [NOICONS] [PUBSCREEN* <Public-Screen>]

Die hinter DRAWER, PATTERN und FILE angegebenen Texte erscheinen als Voreinstellung in den entsprechenden Feldern der Dialogtafel. Hinter POSITIVE und NEGATIVE kommen die Texte für die Symbolschalter OK und Cancel. Das Namensmuster hinter ACCEPTPATTERN bestimmt, welche Dateinamen REQUESTFILE anzeigen soll, das hinter REJECTPATTERN welche nicht.

SAVEMODE sorgt für ein andersfarbiges Listenfenster – damit könnte man eine Auswahl zur Speicherung kennzeichnen. MULTISELECT ermöglicht die Auswahl mehrerer Einträge.

Mit DRAWERSONLY zeigt REQUESTFILE nur Verzeichnisse an – das Eingabefeld für Dateien fehlt. Der Schalter NOICONS schließlich sorgt dafür, daß keine Dateien mit ».info« am Namensende angezeigt werden. Beispiel:

```
requestfile sounds: acceptpatt
ern #?.raw positive "Sound lad
en" negative "Abbruch"
```

RESIDENT Interner Shell-Befehl; macht ein Programm resident, d.h. lädt und verankert es so im Speicher, daß es für alle weiteren Ausführungen nicht mehr geladen werden braucht, oder zeigt eine Liste aller residenten Programme sowie internen Systembefehle an. *Befehlsformat: RESIDENT [<residenter Name>] [<Datei>] [REMOVE] [ADD] [REPLACE] [PURE|FORCE] [SYSTEM]*

REPLACE ist optionales Schlüsselwort für die Standardfunktion: Die angegebene Programmdatei wird geladen und ersetzt eine eventuell vorher residente gemachte (alte) Version. ADD dagegen macht das Programm erneut resident. In diesem Fall muß – bei REPLACE kann – ein anderer Resident-Name vergeben werden. Beispiel:

```
resident test1 df0:testprog
resident test2 df0:testprog
```

Mit REMOVE entfernt RESIDENT (auch interne) Programme aus dem Speicher. Software, die mit der Option SYSTEM verankert wurde, erscheint weder in der mit RESIDENT (ohne Argumente) angeforderten Liste residenter Programme, noch läßt sie sich mit REMOVE entfernen.

Residente Programme müssen -> wiedereintrittsfähig sein, also einen bestimmten Aufbau besitzen. Ist das der Fall (und nur dann) sollte ihnen mit -> PROTECT das »Prädikat« Pure (-> Schutzbit) verliehen werden. RESIDENT mit dem Zusatz PURE oder FORCE verankert auch Programme, deren Pure-Bit nicht gesetzt ist. Das Risiko – fehlerhaftes Arbeiten bis hin zum Programm- oder Systemabsturz – trägt der Anwender. (-> LIST)

Result2 -> Umgebungsvariable der Shell; enthält die Nummer des bei Ausführung des zuletzt aufgerufenen Programms aufgetretenen DOS-Fehlers, oder Null, wenn kein Fehler aufgetreten ist.

Rexxc Systemverzeichnis (Workbench3.0:) mit ARexx-Systemprogrammen

REXXMAST Shell-Befehl bzw. Systemprogramm (Pfad: Workbench3.0:System); startet die Interprozeßkommunikationssprache ARexx.

root directory -> Hauptverzeichnis

Root-Block Spezieller Bereich eines -> Datenträgers für bestimmte Informationen (Art des verwendeten -> Dateisystems) oder ein Programm, das beim -> Neustart geladen und ausgeführt wird. -> INSTALL schreibt die Systemdaten in den Boot-Block und macht den Datenträger damit zur Start- oder Boot-Diskette bzw. -partition.

Rückgabewert Von einem System- oder Anwendungsprogramm nach dem Ablauf übergebener Wert, der als Fehlermeldung interpretiert werden kann, wenn er größer Null ist. Eine besondere Rolle spielen dabei die Werte 5, 10, und 20 für nicht näher bestimmte leichte, mittlere und schwere Fehler. (-> IF, -> DOS, Fehlermeldung)

RUN Interner Systembefehl; startet ein Programm oder mehrere hintereinander so, daß sie im Hintergrund ablaufen. Damit wird kein Ein-/Ausgabefenster geöffnet und die Shell ist sofort wieder frei für weitere Eingaben. *Befehlsformat: RUN <Befehl> [{+ <Befehl>}]*

Ausgabefenster für das Programm im Hintergrund ist das Fenster der aufrufenden Shell, das deshalb auch in der Regel nicht geschlossen werden kann, bevor der neue -> Prozeß beendet ist. Beispiel:

```
run copy brief prt: +
delete brief +
echo "Druck beendet"
```

Sollen mehrere Programme über eine Anweisung aufgerufen werden, sind deren Namen samt dazugehöriger Argumente durch »+« zu separieren. Dabei kann sich das Trennzeichen auch am Ende der Zeile befinden. RUN holt sich die fehlenden Angaben aus der nächsten Zeile.

RX ARexx-Befehl (Workbench3.0:Rexxc); startet das angegebene ARexx-Programm. *Befehlsformat: RX* <Programm> [Argumente]

Fehlt die Pfadangabe, sucht RX das Programm auf dem -> logischen Datenträger REXX (standardmäßig SYS:S).

RXC ARexx-Befehl (Workbench3.0:Rexxc); schließt den residenten ARexx-Interpreter. *Befehlsformat: RXC*

Der Interpreter wird erst dann aus dem Speicher entfernt, wenn das letzte ARexx-Programm abgelaufen ist.

RXLIB ARexx-Befehl (Workbench3.0:Rexxc)

RXSET ARexx-Befehl (Workbench3.0:Rexxc); nimmt ein aus Name und Wert bestehendes Paar in die Clipliste des ARexx-

Das Shell-Glossar begann im AMIGA-Magazin 2/93. Es soll vor allem Besitzer des Amiga 1200, die keine DOS-Dokumentation besitzen, über Funktion und Arbeitsweise der Shell informieren. Als Nachschlagewerk hilft das Shell-Glossar natürlich allen Amiga-Fans. Mit unseren Tips & Tricks zum Betriebssystem (Seite 120 in dieser Ausgabe, Seite 136 in Ausgabe 7/93, Seite 122 in Ausgabe 4/93) geht's dann in die Praxis.

Interpreters auf. *Befehlsformat: RXSET [Name [[=] Zeichenkette]]*

Gibt es dort bereits ein gleichnamiges Paar, wird dies ersetzt. Fehlt der Wert, löscht RXSET das entsprechende Paar der Clipliste. Der Befehl ohne Argumente zeigt alle Paare der Liste an. Beispiel:

```
rxset faktor = 0.75
```

S 1) Systemverzeichnis (Workbench3.0:) für Kommandofolgen.

2) -> logischer Datenträger, der vor Ablauf der -> Startup-Sequence dem Verzeichnis SYS:S zugeordnet wird. -> EXECUTE sucht dort Kommandofolgen, wenn kein Pfad angegeben wurde.

S

Schlüsselwort Kennwort, das angegeben werden muß, um die damit verbundene Funktion auszulösen (»Sesam öffne dich«) (-> Parameter).

Schublade Repräsentation eines Verzeichnisses auf der Workbench.

Schutzbit -> Dateiattribut, das über Art und Zustand einer Datei informiert bzw. bestimmte Operationen damit verhindert. Schutzbits werden mit dem Shell-Befehl -> PROTECT gesetzt und mit -> LIST angezeigt:

- r: Datei kann gelesen werden (readable)
- w: Datei kann beschrieben bzw. geändert werden (writable)
- e: Datei ist ein Programm und damit ausführbar (executable)
- d: Datei kann gelöscht werden (deletable)
- s: Datei ist eine Kommandofolge (script)
- p: Programm in der Datei ist so aufgebaut (pure), daß es im Speicher resident verankert, und damit ohne erneutes Laden mehrmals aufgerufen werden kann (-> RESIDENT)
- a: Datei ist archiviert (archived) worden

ScreenMode Shell-Befehl bzw. -> Voreinstellungs-Editor (Extras3.0:Prefs); öffnet die Dialogtafel zur Einstellung der Bildschirmmodi bzw. übernimmt die Werte einer vorher gespeicherten Datei. *Befehlsformat: SCREENMODE [FROM <Datei>] [EDIT] [USE] [SAVE]*

script file -> Kommandofolge

SEARCH Shell-Befehl bzw. Systemprogramm; sucht eine Zeichenfolge in Dateinamen bzw. in den Dateien des angegebenen Verzeichnisses (einschl. der Unterverzeichnisse). *Befehlsformat: SEARCH [FROM] <Name|Namensmuster> [SEARCH|NAME] <Zeichenfolge|Namensmuster> [ALL] [NONUM] [QUIET] [QUICK] [FILE] [PATTERN]*

Bei Angabe von ALL wird nicht nur das zuerst bzw. hinter FROM angegebene Verzeichnis durchsucht, sondern auch dessen Unterverzeichnisse. Mit QUIET gibt SEARCH nicht den Namen der gerade durchsuchten Datei aus, mit NONUM keine Zeilennummern und mit QUICK verwendet das Systemprogramm ein kompaktes Ausgabeformat. FILE ist anzugeben, wenn SEARCH nicht in den Dateien, sondern in deren Namen suchen soll. PATTERN besagt, daß der Suchbegriff ein -> Namensmuster ist, also Platzhalter wie »#?« enthält.

SEARCH liefert den -> Rückgabewert 0, wenn der Begriff gefunden wurde, und 5 wenn nicht. <Ctrl D> bricht die Suche in der aktuellen Datei ab, nach <Ctrl C> endet der gesamte Suchlauf.

SER -> DOS-Gerät für die unkonvertierte Datenausgabe an der seriellen Schnittstelle, wo meist ein Drucker oder Modem angeschlossen ist.

SERIAL Shell-Befehl bzw. -> Voreinstellungs-Editor (Pfad: Extras3.0:Prefs); öffnet die Dialogtafel zur Einstellung der seriellen Schnittstelle bzw. übernimmt die Werte einer vorher gespeicherten Datei. *Befehlsformat: SERIAL [FROM <Dateiname>] [EDIT] [USE] [SAVE] [PUBSCREEN* <Public-Schirm>] [UNIT <Einheit>]

serial.device Exec-Treiber für die Datenaus- und -eingabe an der seriellen Schnittstelle.

SET Interner Shell-Befehl; setzt eine lokale -> Umgebungsvariable oder zeigt (bei Ausführung ohne Argumente) den Inhalt aller lokalen Variablen an. *Befehlsformat: SET [<Name>] [<Zeichenfolge>]* Beispiel:

```
set Zähler 10
```

(-> GET, -> UNSET)

SETCLOCK Shell-Befehl bzw. Systemprogramm; setzt die batteriegepufferte Echtzeituhr auf die -> Systemzeit (SAVE), die Systemzeit auf die Echtzeit (LOAD) oder setzt die Echtzeituhr zurück. *Befehlsformat: SETCLOCK LOAD|SAVE|RESET* Beispiel:

```
setclock save
```

SETDATE Shell-Befehl bzw. Systemprogramm; setzt den Datums- und Zeitvermerk (Herstellung/letzte Änderung) eines Verzeichnisses bzw. einer oder mehrerer Dateien. *Befehlsformat: SETDATE <Datei|Namensmuster> [<Datum>] [<Uhrzeit>] [ALL]*

Fehlt <Datum> bzw. <Uhrzeit> verwendet SETDATE die Systemzeit. Beispiel:

```
setdate Brief
setdate Mahnung 19-06-93 23:55
```

SETENV Interner Shell-Befehl; weist einer globalen -> Umgebungsvariable einen Wert (Zeichenfolge) zu, oder zeigt (bei Eingabe ohne Argumente) den Inhalt aller globalen Variablen. *Befehlsformat: SETENV [<Name>] [<Zeichenfolge>]* Beispiel:

```
setenv Editor c:CygnusED
```

(-> GETENV, -> UNSETENV)

SETFONT Shell-Befehl bzw. Systemprogramm; setzt den Zeichensatz der aufrufenden Shell. *Befehlsformat: SETFONT <Zeichensatz> <Größe> [SCALE] [PROP] [ITALIC] [BOLD] [UNDERLINE]*

Mit SCALE aktiviert SETFONT die Skalierung von Bitmap-Zeichensätzen, mit PROP werden proportionale Zeichensätze zugelassen. Die restlichen Schalter fordern die entsprechenden Schriftstile an. Beispiel:

```
setfont topaz 13 bold underline
```

SETKEYBOARD Shell-Befehl bzw. Systemprogramm; aktiviert eine landesspezifische Tastaturbelegung. *Befehlsformat: SETKEYBOARD <Tastaturbelegung>*

Die entsprechende Datei mit den Daten muß sich auf dem -> logischen Datenträger KEYMAPS befinden. Beispiel:

```
setmap d
```

SETPATCH Systembefehl; ändert (patcht) den Speicher des Betriebssystems und behebt so Fehler bzw. erweitert diese Routinen. *Befehlsformat: SETPATCH [QUIET] [NOCACHE] [REVERSE]*

QUIET verhindert die Meldungen am Bildschirm, NOCACHE die Aktivierung des Daten-Cache entsprechender Prozessoren und bei REVERSE wird Speicher oberer Adreßbereiche zugewiesen.

Shell Textorientierte -> Bedieneroberfläche des Amiga. Der Systembefehl -> NEWSHELL (bzw. -> NEWCLI) öffnet ein Shell-Fenster und startet ein Programm (-> Prozeß), das über dieses Fenster -> Shell-Anweisungen entgegennimmt. Die Shell ist Nachfolger des CLI, der ersten textorientierten Bedieneroberfläche.

Nach Abschluß einer Shell-Anweisung mit <Return> sucht der Shell-Prozeß das entsprechende Programm zunächst unter den residenten im Speicher. Fehlt es dort, wird eine Datei mit dem angegebenen Namen in den Verzeichnissen gesucht, die in der Suchpfadliste (-> PATH) festgehalten sind, und bei Erfolg geladen. Die Shell startet das Programm und übergibt die beim Aufruf hinter dem Namen angegebenen Parameter zur Auswertung.

Jede Shell verwaltet lokale -> Umgebungsvariablen sowie eine Pseudonym- oder Übersetzungsliste, in die der Shell-Befehl

Shell-Editierfunktionen

<Amiga_rechts C> überträgt im Shell-Fenster markierten Text in die Zwischenablage

<Amiga_rechts V> fügt den Text aus der Zwischenablage an der Cursorposition ein

<Backspace> oder >Ctrl H< löscht das Zeichen links vom Cursor

<Ctrl A> oder <Shift Cursor_links> positioniert den Cursor am Zeilenanfang

<Ctrl J> Zeilenvorschub

<Ctrl K> löscht alle Zeichen von der Cursor-Position bis zum Zeilenende

<Ctrl M> führt die Shell-Anweisung in der Befehlszeile aus (wie <Return>)

<Ctrl U> löscht alle Zeichen von der Cursor-Position bis zum Zeilenanfang

<Ctrl W> löscht das Wort links vom Cursor

<Ctrl X> löscht die aktuelle Zeile

<Ctrl Y> widerruft eine mit <Ctrl K> vorgenommene Löschung

<Ctrl Z> oder <Shift Cursor_rechts> positioniert den Cursor am Zeilenende

<Cursor_oben> blättert vorwärts durch den Befehlszeilenspeicher

<Cursor_unten> blättert rückwärts durch den Befehlszeilenspeicher

<Del> löscht das Zeichen an der Cursor-Position

<Shift Cursor_oben> oder <Ctrl R> durchsucht den Befehlszeilenspeicher rückwärts nach einer Shell-Anweisung, die so beginnt, wie die in der Befehlszeile befindliche, und überträgt die gefundene in die Befehlszeile

<Shift Cursor_unten> durchsucht den Befehlszeilenspeicher vorwärts nach einer Shell-Anweisung, die so beginnt, wie die in der Befehlszeilenbefindliche, und überträgt die gefundene in die Befehlszeile

Weitere Steuerzeichen:

<Backspace> setzt eine mit einem Leerzeichen unterbrochene Ausgabe fort.

<Ctrl \> schließt das Shell-Fenster bzw. stellt bei Verwendung von -> * die normale Befehlszeileneingabe wieder her

<Ctrl C> unterbricht den augenblicklich ablaufenden Befehl

<Ctrl D> unterbricht die augenblickliche ablaufende -> Kommandofolge

<Ctrl Q> setzt eine unterbrochene Ausgabe fort

<Ctrl S> die Ausgabe wird angehalten

Leerzeichen bzw. jedes andere Zeichen hält die laufende Textausgabe an (s. Backspace)

-> ALIAS Abkürzungen für beliebige Zeichenfolgen einträgt. Ab OS 2.0 sind die Systembefehle ALIAS, ASK, CD, ECHO, ELSE, IF, ENDCLI, ENDSHELL, ENDIF, ENDSKIP, FAILAT, FAULT, GETENV, IF, LAB, PATH, PROMPT, QUIT, RESIDENT, SETENV, SKIP, STACK und WHY interne Befehle, also in der Shell integriert, und brauchen deshalb nicht mehr geladen werden.

Alle ausgeführten Anweisungen hält die Shell in einem etwa 2000 Zeichen großen Puffer (-> Befehlszeilenspeicher) fest. Sie lassen sich über die Cursor-Tasten abrufen und damit schnell wiederholen bzw. korrigieren.

Neben <Ctrl \> schließen auch Anklicken des Schalters links oben im Fenster (falls vorhanden) sowie Eingabe von ENDSHELL die Shell. (-> Shell-Startup)

Shell-Anweisung Aufruf eines Anwender- bzw. -> Systemprogramms oder einer -> Befehlsdatei über die -> Shell. Eine Shell-Anweisung beginnt mit dem Namen der Programmdatei, eventuell angeführt vom -> Pfad dorthin. Damit ist festgelegt, was zu tun ist. Die -> Argumente danach bestimmen, womit das zu tun ist und Optionen variieren diese Tätigkeit. Beispiel:

```
dir sys: files
```

»sys:« ist hier das Argument, »files« die Option, die häufig mit »opt« eingeleitet wird. (-> Befehlsschablone)

Argumente und Optionen sind voneinander und vom Befehl durch ein oder mehrere Leerzeichen zu trennen.

Shell-Befehl Name einer in der Shell anzugebenden Systemfunktion bzw. -> Kommandofolge oder Programmdatei. (-> Shell-Anweisung)

Shell-Startup -> Kommandofolge (Workbench:S, S:), die beim Aufruf von NEWSHELL ohne Angabe einer Alternative ausgeführt wird. In die Shell-Startup gehören z.B. ALIAS-Anweisungen, um in allen Shells gültige Abkürzungen (-> Alias) zu definieren.

ShowConfig Shell-Befehl bzw. Systemprogramm (Extras3.0: Tools); gibt Kenndaten des Systems aus (Prozessortyp, Art der Spezialchips, Versionsnummern verschiedener Systemsoftwaremodule, RAM-Ausbau, Erweiterungskarten).

Sicherheitskopie Kopie einer Datei, die angelegt wird für den Fall, daß das Original zerstört, verstümmelt oder sonstwie unbrauchbar wird.

SKIP Interner Shell-Befehl für Befehlsdateien; unterbricht die Ausführung der Kommandofolge so lange, bis er auf die angegebene -> Sprungmarke trifft – alle Anweisungen zwischen SKIP und der Sprungmarke werden also ignoriert. *Befehlsformat: SKIP [<Sprungmarke>] [BACK]*

Bei Angabe von BACK wird die Sprungmarke vor der SKIP-Anweisung gesucht. Ein Rücksprung ist nur möglich, wenn sich zwischen Sprungmarke und SKIP-Anweisung kein EXECUTE befindet. Beispiele:

```
skip loop
skip loop back
```

Mit Skip und Umgebungsvariablen lassen sich Schleifen programmieren. (-> LAB, -> EVAL)

Skript -> Kommandofolge

SORT Shell-Befehl bzw. Systemprogramm; sortiert die hinter FROM angegebene Datei und speichert das Ergebnis in die hinter TO angegebene. *Befehlsformat: SORT [FROM] <Datei/Namensmuster> [TO] <Datei> [COLSTART <n>] [CASE] [NUMERIC]*

Die Zeilen der zu sortierenden Datei müssen mit einem Wagenrücklauf (ASCII 13) oder Zeilenvorschubzeichen (10) enden. CASE sorgt dafür, daß Groß- vor Kleinbuchstaben einsortiert werde, NUMERIC, daß nur Ziffern am Anfang des Sortierbegriffs berücksichtigt werden (keine Ziffer = 0). Mit COLSTART läßt sich bestimmen, ab dem wievielten Zeichen jeder Zeile der Sortierbegriff beginnt. Beispiel:

```
sort Lexikon Lex.sort alpha
```

Sound (Extras3.0:Prefs) Shell-Befehl bzw. -> Voreinstellungs-Editor (Extras3.0:Prefs); öffnet eine Dialogtafel zur Einstellung der Signaltonparameter bzw. übernimmt diese einer vorher gespeicherten Datei. *Befehlsformat: SOUND [FROM <Datei>] [EDIT] [USE] [SAVE] [PUBSCREEN <Public-Screen>]*

sound.datatype Importtreiber zum Laden digitaler Sounddaten.

SPAT -> Kommandofolge (Workbench3.0:s, S:) ermöglicht die Angabe von Namensmustern bei Systembefehlen, die dafür nicht vorgesehen sind. Im Gegensatz zu DPAT (double pattern) wird SPAT (single pattern) bei Systembefehlen mit einem Argument verwendet.

Sprungmarke Name, der als Zielangabe eines Sprungbefehls dient. In Kommandofolgen werden Sprungmarken mit -> LAB definiert und mit -> SKIP angesprungen. Die Anweisungen zwischen den betreffenden SKIP und LAB werden nicht ausgeführt.

STACK Interner Shell-Befehl; legt die Größe des -> Stapelspeichers fest, den die Shell, in der dieser Befehl ausgeführt wurde, danach aufgerufenen Programmen übergibt bzw. zeigt die Größe bei Aufruf ohne Argumente an. *Befehlsformat: STACK [<n>]* Beispiel:

```
stack 10000
```

Standardprogramm (default tool) In einer Projektpiktogrammdatei festgehaltener Name des Programms, das das zugehörige Projekt gespeichert hat. Die nach Aufruf der Menüfunktion »Icons/Information« erscheinende Dialogtafel zeigt diesen Namen, der dort auch verändert werden kann. Nach einem Doppelklick auf das Projektpiktogramm lädt und startet die Workbench das Standardprogramm, was wiederum das angeklickte Projekt lädt.

Startdiskette -> Datenträger mit einem besonderen Kennzeichen, das ihn als Datenträger mit bestimmten Systemdaten (Workbench) identifiziert. Der Shell-Befehl -> INSTALL bringt dieses Kennzeichen auf. Dabei wird nicht geprüft, ob die Systemdaten auch vorhanden sind. Die Startdiskette ist für den Start (das Booten) des Betriebssystems nicht erforderlich, wenn sich die Systemdaten auf einer autobootfähigen Festplatte befinden.

startup-sequence -> Kommandofolge (SYS:S, S:); die darin befindlichen Shell-Anweisungen werden beim Start (Einschalten oder -> Reset) des Computers ausgeführt.

STATUS Shell-Befehl bzw. Systemprogramm; gibt Informationen über Shell-Prozesse aus. *Befehlsformat: STATUS [<Prozeß>] [FULL] [TCB] [CLI|ALL] [Command <Befehl>]*

Ohne Angabe von <Prozeß> oder mit ALL gibt STATUS die angeforderten Informationen zu allen Prozessen aus. Über TCB und FULL werden auch Angaben zur Stackgröße, Priorität und zu globalen Vektoren angefordert.

Mit COMMAND sucht STATUS nach einem aktiven Befehl oder Programm und gibt – falls gefunden – die entsprechende Nummer des Shell-Prozesses aus. Beispiele:

```
status
status 1 full
status >ram:$$$ command=copy
break <ram:$$$
```

Die beiden letzten Anweisungen brechen einen aktiven COPY ab, ganz gleich, von welcher Shell er gestartet wurde.

Statusbit -> Schutzbit

Storage3.0 Systemdiskette; alle Verzeichnisse dieser Diskette enthalten Systemdaten (Mounteinträge, -> Tastaturtabellen, Monitorparameter, -> Druckertreiber). Die Dateien werden bei der Installation des Systems auf Festplatte je nach Bedarf oder insgesamt in die gleichnamigen, bereits von der -> Workbench installierten Verzeichnisse kopiert.

Suchpfad (search path) Liste von Verzeichnissen, in denen die Shell Programme sucht, bei deren Aufruf kein Pfadname angegeben wurde. Die Suche beginnt im aktuellen Verzeichnis, dann kommen alle mit dem Shell-Befehl -> PATH ergänzten Verzeichnisse dran, und wenn das Programm noch immer nicht gefunden ist, schaut die Shell zum Schluß auf dem -> logischen Datenträger C nach.

Anweisungen der Standard-Startup-Sequence ergänzen den Suchpfad mit den Verzeichnissen Utilities, Prefs, S und der -> RAM-Disk. PATH ohne Angabe eines Verzeichnisses gibt den Suchpfad aus. Er gilt für die Shell, über deren Fenster er festgelegt wurde, sowie alle darüber gestarteten Shells.

SYS logischer Datenträger, der vor Ablauf der Startup-Sequence dem Wurzelverzeichnis der Startdiskette bzw. der entsprechenden Boot-Partition auf Festplatte zugeordnet wird. Über den Pfad SYS: kann man auf die Systemverzeichnisse zugreifen, ohne den Namen der Workbench-Diskette oder System-Partition zu kennen.

Sys Systemverzeichnis (Workbench3.0:Prefs/Env-Archive) mit den Systemeinstellungen (wie Bildschirmauflösung und -farben, Druckertreiber, Workbench-Muster).

System Systemverzeichnis (Workbench3.0:, Extras3.0:) mit den Systemprogrammen CLI, DISKCOPY, FIXFONTS, FORMAT, NOFASTMEM und REXXMAST.

system-configuration Datei (Workbench3.0:devs, DEVS) mit den Systemeinstellungen. Ab OS 2.0 wird diese Datei nur noch unterstützt, weil einige ältere Programme sie erwarten.

Systemanweisung -> Shell-Anweisung

Fortsetzung in der nächsten Ausgabe