Introduction

Mid 1980's the Sinclair QL arrived with new storage devices **microdrive** '**mdv1_** & **mdv2_**'. It wasn't long before external **floppy drives** '**flp1_**' & '**flip2_**'were available and **Hard Drives** '**win1**_' etc, followed soon after to open up the possibility of even larger storage capacities

The **QL Technical Guide** identified the **QL Filename** as being up to **36 Bytes** in length or the equivalent in **ASCII Characters.** Viewing files using the SuperBASIC **DIR** command displays a single vertical list. This soon spread over several screens, mistyping a file name became a frustrating exercise reviewing the spelling with the DIR command. The **QBITS** approach was to develop a more friendly **File Directory Handler**.

QBITS File Management Concepts

By 1987 a collection of SuperBASIC routines to keep track and review Filenames evolved into an early File Management Tool called File Tidy later shortened to **FTidy**. It accessed a Source Device and viewed up to 160 Filenames. **QBITS FTidy** was submitted to QUANTA a short while before Howard Clase published his **FTidy** program in QLWorld Sep1988.

QBITS FTidy128

The screen displayed four columns of file names of up to 18 characters. However, the full Filename of up to 36 characters, when selected was shown in the window below the Menu bar. **COPY** and **DELETE** commands allowed for single File or batch processing of multiple files. **SelDev** was used for **Source** and **Target** selection. The **Print** command was for Printer export of File lists. These were later dropped for **LOAD**/**LRUN**.

The **FDIR** with **SubDIR**ectories were added with Millennium updates. The latest release now includes a **WildCard** option, **VIEW** with ASCII or HEX Readout, **EXEC** for **_obj** files plus **ZIP** to Load and Compile a SuperBASIC **_**bas File.

QBITS FTidySE

Screen layout is divided into four areas. Top centre the Title Box displays **QBITS FTidy**+ at start up and prompts for Selection of a Source device shown in the bottom action window.

The main display area shows a **Help** screen of Commands with a brief description of their functions. Navigation of Menu Commands and Displayed Filenames is by Cursor Keys ← ↑ ↓ → with actions taken by ▬ Spacebar ↵Enter Keys. The Menu can also be Selected by individual Keys **F M S C D O L R V Z**. For Help screen Press 'I' for Info ⓘ, Press 'Q' to Quit ⏻ the program.

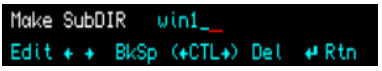
QBITS FTidySE File Directory

FDIR displays the **DIRectory** of the Default or last **Device** chosen. Full Filenames and Stats: Byte Size & Date/Time Stamp Entries of the Selected Storage Device are copied to **FList**. The Filenames are then selected sequentially by a **FOR loop** and compared within a **REPeat Loop** with the output **DFile$(n)** array sorted in Alphanumeric order. A second **FOR loop** selects any **SubDIR**ectory names and lists them before the **Filenames**.

The File DIRectory [**FDIR**] or SubDIRirectories [**SDIR**] if present, are displayed under **Volume** the **Device name, Volume/Sectors** and under **Files** the **Number** held in Directory. The Filenames of the selected **Device DIR**ectory or **SubDIR**ectory are Read and Sorted to generate the new display of Filenames. Any selected **Filename** is shown in full with Stats **Bytes** size and **Date/Time** Stamp in the lower action window.

QBITS FTidySE MDIR

Use Line Editor to create a new **SubD**irectory. QBITS six level **SubDIR** allocation…

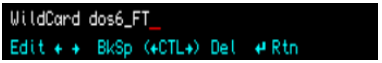


DIM SubDIR$(6,24) where SubDIR$(1) = "**SD1**_" and SubDIR$(6) access = "**SD1_SD2_SD3_SD4_SD5_SD6**_"

QBITS FTidySE SubDIRectories

The full identity of a file location can be 41 Characters. This begins with the Drive Device a five-character identifier, ie. mdv1_, flp1_, win1_, the fifth character being '**_**' an underscore. The next 36 Characters make up the Filename, with the first twenty-four characters considered for use as **SubDIR**ectory's. For example, 'SubDIR1_' which as with Drive names ends with an underscore. If they were named with letters of the alphabet, 'A_' to 'L_' we could potentially create twelve **SubDIR**rectory levels. **QBITS FTidy** limit is set at six **SubDIR** levels.

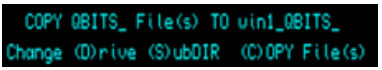
QBITS FTidySE WildCard



Press '**#**' and Type WildCard Characters to Search a Filename or Subgroup of Files. If no matches are found DEVICE ERROR is displayed. Restore List with FDIR or SDIR.

QBITS FTidySE COPY / DELETE

Select single or multiple files. The Filename(s) are identified by moving through the files listed and highlighting with the Spacebar. For **COPY** select a destination **Target** device with (**D**)rive and (**S**)ubDIR.



Press (**C**)opy and selected Filename(s) from **Source DIR**ectory are then copied across to the **Target** device. Any files of same name in destination device prompts for further action with an **Overwrite 'y/n'**.

Similarly, for DELETE a highlighted Filename is confirmed with '**y/n**' before Deleting.

QBITS FTidySE EXEC / LRUN

Select a Filename, the full Filename up to 36 characters is displayed together with its Byte length and Time/Date Stamp. **EXEC** is for Object files, machine code files that can be loaded and run as an independent JOB. **LRUN** is for S/SuperBASIC program that Load and Run under the QL Interpreter. For both you are prompted with '**y/n**' to action.

QBITS FTidySE RENAME

Select an existing Filename (**File$**) and **Edit** the string (**str$**) with the **QBITS Line Editor**. Action with ↵ Enter and the **FTidy** checks that the Filename doesn't already exist. If not a COPY with new filename is made to source and the old file Deleted.

QBITS FTidySE VIEW

View was seen as a useful option in being able to OPEN and read the contents of a File. **S/SuperBASIC** Files are in **Plain Text**, others can be **Data** lists or a mix of code blocks and ASCII characters as in **Executable** machine code. **FTidy View** gives the option to display as **ASCII characters (y)** or in **Hex Bytes (Enter).**

QBITS FTidySE ZIP

Compiler **Q_LIBERATOR** needs to be installed, **QLib_RUN** if installed can be shared by programs or it can be linked in as part of the Compile process. Select a **SupeBASIC** File with a **'_bas'** suffix. A Prompt (y/n Enter) appears. Press '**y**' if **RUN**time is resident or '**Enter**' if to be linked with program. Pressing 'n' returns to **FTidy Menu** bar.

When completed return to **QBITS FTidySE** with **ALT-f** and activate the Compiled Program. **w**ith **EXEC Filename_obj**

QBITS FTidySE use of **LIBERATOR**
An early notion was to include a link to Compile SuperBASIC Programs. In computing, ZIP is a term used for File(s) or Routines compressed into a package and descriptively can also imply to go faster. A Spectrum BASIC Compiler written by N Goodwin was called ZIP. I hope he won't object to QBITS use of the word!

Attempting to maintain some compatibility across various QL Platforms, the compiler choice was between SUPERCHARGE, TURBO or LIBERATOR. For the present QBITS_FTidySE relies on the SMSQ/E O/S being in place and use of Sinclair QL Forum Edition 2020 for QPC2 arrangement of QLIBERATOR. Users should refer to the User Manuals for a better understanding of limitations with other environments.

QBITS FTidySE and **Handling of Screen Positioning**
Filenames are listed across the screen in columns and rows, each column having a defined number of characters or string length. If a name exceeds this length, then the name is truncated. If the name string is less, the missing characters are filled with spaces to overwrite the possibility of characters left over from a previously displayed Filename. If a Filename List exceeds the available screen space, upon reaching the bottom of the window the bottom row can be scrolled up and a new group of filenames added. To keep track a Line pointer identifies the position in the Filename list and a Row pointer for the screen position. A filename position can be identified by subtracting the Row pointer from the File Line pointer. A similar arrangement is utilised for scrolling the screen down so as to add rows at the screen top.

Note: A program having run its course needs to close down, release RAM etc and restore the system back to previous settings

Note: File & DIRectory Info
QL Files appear as an array of bytes on a physical storage device such as microdrives - mdv1_, Floppy disks - flp1_ and Hard drives - win1_ etc. The storage system is composed of 512-byte blocks, addressing is via an associated File Pointer by Block number (sector) and Byte number within that block. The QL Tech Guide describes the 64 Bytes File header as follows:

$00	long	file length
$04	byte	file access key (not yet implemented - set at 0)
$05	byte	file type
$06	8bytes	file type-dependant information
$0E	2+36 Bytes	filename
$34	long	reserved for update Date (not yet implemented)
$38	long	reserved for reference Date (not yet implemented)
$3C	long	reserved for backup Date (not yet implemented)

The file types allowed at the time of original QL:

2 a relocatable object file
1 an executable program where the first longword of the type-dependant information holds the default size of data space required for the program.
0 for anything else

QBITS FTidySE Procedures

Init_win	Initialises Program setting the screen display
F_Title	Displays QBITS Title / DIR headings
F_Info	Displays Info/Help screen
Cmd_Menu	Main Program Loop
SelDrv	Selects Source or Target Devices
FileDIR	Generates File List of Device DIRectory or a Sub DIRectory
F_Sort	Arranges Filenames AlphaNumerically with SubDIR(s) first
MakeDIR	Uses Line Editor to create a New SubDIRectory
SubDIR	Selects and displays the Filenames of a SubDIRectory
Sub_up	Move to Drive DIR or Higher-Level SubDIR
Sub_dn	Move to a Lower-Level SubDIR
WildCard	File Group Select with WildCard Charcters.
F_Select	Use Cursor keys to Select a SubDIR or Filename
Fscr_posn	Calculate screen position of SubDIR or Filename
Fscr_up	Scroll up one row
Fscr_dn	Scroll down one row
F_write	Print SubDIR or Filename and Stats o screen position
F_clear	Prints updated File List to screen
F_Chk	Returns y/n Enter answer [no=0 yes=1 Enter=2
F_Copy	Selects and Confirms Filename(s) y/n? from Source Device
F_Target	Selects destination - Targeted Device DIR/SubDIR
F_Copy2	Copies File(s) to Target Device DIR SubDIR with overwrite y/n?
F_Delete	Deletes Selected Files(s) from Source Device y/n?
F_Lrun(act)	EXEC or Load/RUN selected File from displayed list
F_Rename	Uses Line editor to Rename a selected Filename
F_View	Prints ASCII or Hex Code of selected Filename to screen
F_ZIP	Selected SuperBASIC File linked into QLIBERATOR Compiler
Ln_Ed	Line Editor action menu
Str_chk	Checks if last Character of string is '**_**' and deletes
Ln_Prn	Prints Filename to Screen
Ln_Cur	Prints Current Cursor Position to Screen
Add_chr	Adds a character anywhere within String or at end.
Del_chr	Deletes a character anywhere in string
KInfo	Key Graphic Image created for Info/Help
KQuit	Key Graphic Image created for Quit
GDrive	Hard Drive symbol
GFolder	File Folder Symbol

QBITS FTidySE_bas Code

```
1000 REMark QBITS_FTidySE_bas [QBITS FTidySE 2024 QL40th - QPC2] vM30

1002 dev$='win1_':MODE 4:gx=0:gy=0    :REMark Basic Settings

1004 WHEN ERRor :eck=1:CONTINUE:END WHEN
```

Note: **dev$** is set to the default source device which is accessed to Import **QBITSConfig** settings, if nor found the program will hang. If FDIR fails to find a Device or Files then it returns **DEVICE ERROR** and instead of the Interpreter halting the program WHEN ERRor will CONTINUE the Program.

```
1006 REMark Import QBITSConfig Settings - QPC2
1007 OPEN_IN#9,dev$&'QBITSConfig':INPUT#9,gx\gy\dn$\dev$\dn%\dm%
1008 DIM Drv$(dm%,5):FOR d=0 TO dm%:INPUT#9,Drv$(d):END FOR d:CLOSE#9

1010 REMark Set max% number of Files :DIM Arrays
1011 max%=300
1012 DIM DFile$(max%,2,36),fink%(max%),CFile$(max%,2,36),cink%(max%)
1013 DIM Comd$(58),key$(3,52),help$(9,48),Time$(20)
1014 DIM DD$(5),DDIR$(24),SD$(5),SDIR$(24),TD$(5),TDIR$(24),str$(36)
```

Note: DFile$(max%,2,36)
Filename=1
Stats =2
Max ASCII Characters

```
1016 REMark FTidy Setup & Comand Menu

1018 Init_win:Cmd_Menu

1020 DEFine PROCedure Init_win
1021 OPEN#5,scr_:WINDOW#5,512,256,gx,gy          :PAPER#5,0:BORDER#5,1,3:CLS#5
1022 OPEN#4,scr_:WINDOW#4,504,214,gx+4,gy+7      :PAPER#4,0:BORDER#4,1,5:CSIZE#4,1,0
1023 OPEN#3,scr_:WINDOW#3,280,26,gx+114,gy+2     :PAPER#3,0:BORDER#3,1,5:CLS#3
1024 WINDOW#2,504,220,gx+4,gy+2 :PAPER#2,0                  :CSIZE#2,0,0:INK#2,7
1025 WINDOW#1,496,162,gx+8,gy+42:PAPER#1,0       :BORDER#1,1,6:CSIZE#1,0,0:INK#1,7
1026 WINDOW#0,512, 34,gx,gy+222 :PAPER#0,0       :BORDER#0,1,3:CSIZE#0,0,0:INK#0,7
1027 OVER#4,1:INK#4,3
1028 FOR i=0 TO 1:CURSOR#4,  4+i,12:PRINT#4,'VOLUME'
1029 FOR i=0 TO 1:CURSOR#4,450+i,12:PRINT#4,'/FILES'
1030 OVER#4,0:INK#2,6:Time$=DATE$
1031 CURSOR#2,6,7:PRINT#2,Time$(10 TO 11)&Time$(5 TO 9)&Time$(1 TO 4)
1032 SCALE#2,100,0,0:SCALE#1,100,0,0:F_Title 2,66,'QBITS FTidy':F_Info
1033 END DEFine
```


```
1035 DEFine PROCedure F_Title (cs%,x%,Title$)
1036 CLS#3:CSIZE#3,cs%,1:OVER#3,1
1037 INK#3,2 :FOR i=0 TO 1:CURSOR#3,x%+i,1:PRINT#3,Title$
1038 INK#3,6 :FOR i=1 TO 2:CURSOR#3,x%+i,2:PRINT#3,Title$:OVER#3,0
1039 END DEFine
```

Note: cs% - Character SIZE: x% - horizontal offset: Title$ - Character string.

```
1041 DEFine PROCedure F_Info
1042 help$(0)=' FDIR       - Show FileDIRectory on Source Device'
1043 help$(1)=' MDIR       - Make SubDIRectory  on Source Device'
1044 help$(2)=' SDIR       - Show SubDIRectory  on Source Device'
1045 help$(3)=' COPY       - File(s)from Source TO Target Device'
1046 help$(4)='DELETE      - File(s)from Source Device'
1047 help$(5)=' EXEC       - EXEC Selected Object File'
1048 help$(6)=' LRUN       - LRUN Selected SuperBASIC File'
1049 help$(7)='RENAME      - Edit Name of Selected File'
1050 help$(8)=' VIEW       - ASCII String(y) or HEX(Enter)'
1051 help$(9)=' ZIP        - Load & COMPILE a SuperBASIC File'
1052 OVER#1,1:CSIZE#1,1,0:CLS#1:sp=20
1053 FOR hp=0 TO 8
1054   INK#1,7:FOR i=0 TO 1:CURSOR#1,54+i,sp+hp*10:PRINT#1,help$(hp,1 TO 9);
1055   INK#1,5:PRINT#1,help$(hp,10 TO):IF hp=2:sp=24
1056 END FOR hp
1057 OVER#1,0:CSIZE#1,0,0:INK#1,5:GDrive 7,31,92:GFolder 7,136,91
1058 CURSOR#1,106,  8:PRINT#1,'Select Menu with ← → CURSOR keys and ← ENTER'
1059 CURSOR#1,106,128:PRINT#1,'Select File(s)  ← ↑ ↓ → Mark    Page Up/Down ALT ↑ ↓'
1060 BLOCK#1,2,4,334,10,5:BLOCK#1,12,3,256,132,5:INK#1,6:px%=108:DD$=''
1061 CURSOR#1,132,142:PRINT#1,"[#] WildCard     (I)nfo Panel     (Q)uit"
1062 KInfo 1,6,94,8,2.5:KQuit 1,6,167,8,2.5: cmd=1:mm%=1:mx%=9:lptr%=0:INK#0,7
1063 END DEFine
```


```
1065 DEFine PROCedure Cmd_Menu
1066 DIM cx%(9),Cmd$(10,6):RESTORE 1067:FOR i=1 TO 10:READ cx%(i),Cmd$(i)
1067 DATA 16,'FDIR',50,'MDIR',90,'SDIR',144,'COPY',182,'DELETE'
1068 DATA 240,'EXEC',280,'LRUN',320,'RENAME',376,'VIEW',416,'ZIP'
1069 STRIP#4,0:INK#4,7:FOR i=1 TO 10:CURSOR#4,cx%(i),200:PRINT#4,Cmd$(i)
1070 KInfo 2,7,158,4.6,2:KExit 2,7,164,4.4,2:SelDrv:CLS#0 :WC$='':OW$=WC$
1071 REPeat Cmd_lp
1072   STRIP#4,6:INK#4,0:CURSOR#4,cx%(cmd),200:PRINT#4,Cmd$(cmd)
1073   k$=(INKEY$(#0,-1)):k=CODE(k$)
1074   STRIP#4,0:INK#4,7:CURSOR#4,cx%(cmd),200:PRINT#4,Cmd$(cmd)
1075   SELect ON k
1076     =192:cmd=cmd -1:IF cmd<1   :cmd=10    :REMark← Left   Cursor min-max
1077     =200:cmd=cmd+1:IF cmd>10 :cmd=1      :REMark →Right Cursor max=min
1078     =73,105:F_Info   :PAUSE :CLS:F_clear      :CLS#0      :REMark (I)nfo
1079     =81,113:IF dn$='':STOP   :ELSE LRUN dn$ :STOP      :REMark(Q)uit
1080     =35       :WildCard:CLS#0:OW$=WC$:WC$=''          :REMark (#) files
1081     =10,65 TO 122:IF k<>10:cmd=1+(k$ INSTR 'FfMmSsCcDdEeLlRrVvZz')DIV 2
1082       SELect ON cmd
1083         =  1:SelDrv                :CLS#0
1084         =  2:MakeDIR               :CLS#0
1085         =  3:SubDIR                :CLS#0
1086         =  4:IF ftot%>0:F_Copy     :CLS#0
1087         =  5:IF ftot%>0:F_Delete   :CLS#0
1088         =  6:IF ftot%>0:F_Run 2    :CLS#0   :REMark EXEC
1089         =  7:IF ftot%>0:F_Run 1    :CLS#0   :REMark LRUN
1090         =  8:IF ftot%>0:F_Rename   :CLS#0
1091         =  9:IF ftot%>0:F_View     :CLS#0   :REMark ASCII(y) HEX(Entr)
1092         =10:IF ftot%>0:F_ZIP       :CLS#0   :REMark Compiler
1093       END SELect
1094   END SELect
1095 END REPeat Cmd_lp
1096 END DEFine
```

```
1100 REMark QBITS FTidy DIRectory Management

1102 DEFine PROCedure SelDrv
1103 DIM SubDIR$(6,24):dl%=0:OD$=DD$:dch=0
1104 IF cmd=1:INK#0,7:CURSOR#0,18,6:PRINT#0,'Select Drive: ':px%=108
1105 REPeat Dr_lp
1106   INK#0,5:CURSOR#0,px%,6:PRINT#0,Drv$(dn%)&' (↑↓) <ENTER>':CLS#0,4
1107   k=CODE(INKEY$(#0,-1))
1108   SELect ON k
1109     =10:DD$=Drv$(dn%):EXIT Dr_lp          :REMark Enter Select Drive
1110     =208:dn%=dn%-1:IF dn%< 0:dn%=15       :REMark Up
1111     =216:dn%=dn%+1:IF dn%>15:dn%= 0       :REMark Down
1112   END SELect
1113 END REPeat Dr_lp
1114 IF OD$=DD$ AND OW$=WC$ :RETurn :ELSE DDIR$='':FileDIR
1115 END DEFine
```

Select Drive: win1_ (↑↓) <ENTER>


```
1117 DEFine PROCedure FileDIR
1118 CLS#0:DELETE DD$&DDIR$&'FList':F_Title 1,4,'DIR '&DD$&DDIR$
1119 IF cmd<2:INK#0,5:CURSOR#0,24,6:PRINT#0,'Files being Selected...'
1120 OPEN_NEW#9,DD$&DDIR$&'FList':STAT#9,DD$&DDIR$:WSTAT#9,DD$&DDIR$:CLOSE#9
1121 OPEN_IN #9,DD$&DDIR$&'FList':INPUT#9,DName$\DSec$:n=1:ftot%=0
1122 REPeat DIR_lp
1123   IF EOF(#9) OR n>max%:ftot%=n-1:CLOSE#9:EXIT DIR_lp
1124   INPUT#9,DFile$(n,1):fink%(n)=5:IF WC$ INSTR DFile$(n,1)=0:NEXT Dir_lp
1125   IF '>' INSTR DFile$(n,1)=0:INPUT#9,DFile$(n,2):n=n+1:ELSE n=n+1
1126 END REPeat DIR_lp
1127 BLOCK#2,88,10,414,7,0:IF dl%>0:CURSOR#2,414,7:PRINT#2,'SubDIR Level:';dl%
1128 BLOCK#2,480,10,4,29,0:INK#2,6 :CURSOR#2,4,29:PRINT#2,DName$;'  ';DSec$
1129 CURSOR#2,466,29:PRINT#2,FILL$(' ',5-LEN(ftot%))&ftot%
1130 IF ftot%=0:stot%=0:CLS:F_Title 1,4,'DEVICE ERROR':RETurn
1131 F_Sort:nm%=1:nx%=ftot%:lptr%=0:CLS#1:F_clear:n=1
1132 END DEFine
```

Files being Selected...

SubDIR Level:1 /FILES

QBITSProgs QDOS 88527528/430077944 sectors

```
1134 DEFine PROCedure F_Sort
1135 FOR sn=1 TO ftot%
1136   p=sn:comp$=DFile$(p,1):info$=DFile$(p,2)
1137   REPeat Sort_lp
1138     IF comp$>=DFile$(p-1,1):EXIT Sort_lp
1139     DFile$(p,1)=DFile$(p-1,1):DFile$(p,2)=DFile$(p-1,2):p=p-1
1140   END REPeat Sort_lp
1141   DFile$(p,1)=comp$:DFile$(p,2)=info$
1142 END FOR sn
1143 ntop=1:nsel=1:stot%=0
1144 FOR sn=1 TO ftot%
1145   IF '>' INSTR DFile$(sn,1)
1146     comp$=DFile$(sn,1):info$=DFile$(sn,2):nsel=sn-1
1147     FOR p=nsel TO ntop STEP -1
1148       DFile$(p+1,1)=DFile$(p,1):DFile$(p+1,2)=DFile$(p,2)
1149     END FOR p
1150     DFile$(ntop,1)=comp$:DFile$(ntop,2)=info$:ntop=ntop+1:stot%=stot%+1
1151   END IF
1152 END FOR sn
1153 END DEFine
```


```
1155 DEFine PROCedure MakeDIR
1156 md%=24-LEN(DDIR$):px%=138+LEN(DDIR$)*6
1157 IF md%<3:CURSOR#0,24,6:PRINT#0,'Lowest Level Reached':PAUSE 50:RETurn
1158 INK#0,7 :CURSOR#0,24,6:PRINT#0,'Make SubDIR    ';:INK#0,5:PRINT#0,DD$&DDIR$
1159 cp%=1:sl%=0:sm%=md%:str$='':Ln_Ed:IF LEN(str$)=0:RETurn
1160 CURSOR#0,px%+LEN(str$)*6,6:PRINT#0,' (y/n)':K_Chk
1161 IF chk=1
1162    FOR n=1 TO stot%:IF DDIR$&str$ INSTR DFile$(n,1):RETurn
1163    MAKE_DIR DD$&DDIR$&str$:FileDIR
1164 END IF
1165 END DEFine
```


```
1167 DEFine PROCedure SubDIR
1168 INK#0,7:CURSOR#0,24,6:PRINT#0,'SubDIR ';:INK#0,5:PRINT#0,DD$&DDIR$;
1169 INK#0,7:PRINT#0,' ↑↓ ':CLS#0,4:k=CODE(INKEY$(#0,-1))
1170 IF k=208 AND dl%>=0   :Sub_up          dl% DIRectory Level
1171 IF k=216 AND stot%>=0:Sub_dn          stot% Sub Total
1172 IF k=10 OR k=32:FileDIR
1173 END DEFine

1175 DEFine PROCedure Sub_up
1176 SubDIR$(dl%)='':dl%=dl%-1:DDIR$=SubDIR$(dl%):FileDIR
1177 END DEFine

1179 DEFine PROCedure Sub_dn
1180 IF stot%<1 OR dl%=6:RETurn
1181 px%=96:mark%=5:n=1:nm%=1:nx%=stot%:st%=1:F_select:st%=0
1182 INK#0,5:CURSOR#0,px%,6:PRINT#0,DFile$(n,1)&' (y/n)':CLS#0,4:K_Chk
1183 IF chk=1
1184   DDIR$=DFile$(n,1,1 TO flen%-3)&'_':dl%=dl%+1:nm%=stot%:nx%=ftot%
1185   FileDIR:SubDIR$(dl%)=DDIR$:CURSOR#0,px%+12,6:PRINT#0,DDIR$:CLS#0,4
1186 END IF
1187 nm%=1:nx%=ftot%:F_clear:n=1
1188 END DEFine
```

QBITS FTidy SubDIRectories

The full identity of a file location can be 41 Characters. This begins with the Drive Device a five-character identifier, ie. mdv1_ flp1_ win1_ dos1_ the fifth character '_' always being an underscore. The next 36 Characters make up the Filename, of which the first twenty-four characters can be considered for **SubDIR**ectory use. For example, 'SubDIR1_' which as with Drive names have to end with an underscore. If they were named with letters of the alphabet, 'A_' to 'L_' we could potentially create twelve SubDIR levels. However, QBITS File Tidy is limited this to just SIX Sub levels, further levels will not be accessed by **SDIR.**

DIM SubDIR$(6,24) where SubDIR$(1) = "**SD1**_" and SubDIR$(6) access = "**SD1_SD2_SD3_SD4_SD5_SD6**_"

```
1190 DEFinePROCedure WildCard
1191 CLS#0:INK#0,7:CURSOR#0,24,6:PRINT#0,'WildCard ';:PRINT#0,DD$&DDIR$
1192 cp%=1:sm%=20:INK#0,7:px%=108+LEN(DDIR$)*6:Ln_Ed WC$:FileDIR:CLS#0
1193 END DEFine
```

Note: FDIR & **SDIR** Rests WildCard **WC$="**

```
1200 REMark QBITS FTidy Filename Display

1202 DEFine PROCedure F_select                    filename: select and highlight in screen location
1203 IF cmd>3:INK#0,7:CURSOR#0,24,6:PRINT#0,Cmd$(cmd):CLS#0,4     Note: Check a File Command
1204 INK#0,5:CURSOR#0,66,6:PRINT#0,DD$&DDIR$                          Device & SubDIR
1205 REPeat Sel_lp
1206  Fscr_posn:fink%(n)=7:F_write:fink%(n)=5:k=CODE(INKEY$(#0,-1))
1207  SELect ON k
1208    =192:Fscr_posn:F_write:n=n  -1          :REMark Back      1
1209    =200:Fscr_posn:F_write:n=n +1          :REMark Forward   1
1210    =208:Fscr_posn:F_write:n=n  -4          :REMark Up        1 Row
1211    =216:Fscr_posn:F_write:n=n +4          :REMark Down      1 Row
1212    =209:Fscr_posn:F_write:n=n -60         :REMark Up        1 Page
1213    =217:Fscr_posn:F_write:n=n+60          :REMark Down      1 Page
1214    = 32:fink%(n)=mark%:F_write:n=n+1      :REMark mark Filename   mark% in Highlight Colour
1215    = 10:fink%(n)=7:CURSOR#0,0,20:CLS#0,4:RETurn                   fink%(n) File Ink Highlight Colour
1216  END SELect
1217 END REPeat Sel_lp
1218 END DEFine

1220 DEFine PROCedure Fscr_posn                   calculate screen position
1221 IF n<nm%:n=nm%                              n file number nm% min nx% max
1222 IF n>nx%:n=nx%                          Note: n floating point – necessary with QDOS FOR loops
1223 fptr%=n-1:frow%=(fptr% DIV 4)
1224 IF frow%>(15+lptr%):Fscr_up:n=fptr%+1:Fscr_posn frow% file row fptr% file pointer lptr% line pointer
1225 IF frow%<(  0+lptr%):Fscr_dn:n=fptr%+1:Fscr_posn
1226 srow%=frow%-lptr%:scol%=(fptr% MOD 4)*20        srow% screen row scol% screen column
1227 END DEFine

1229 DEFine PROCedure Fscr_up
1230 lptr%=lptr%+1:SCROLL#1,-10:n=(lptr%+15)*4:srow%=15
1231 FOR i=0 TO 3:scol%=i*20:n=n+1:F_write
1232 END DEFine

1234 DEFine PROCedure Fscr_dn
1235 lptr%=lptr%-1:SCROLL#1,10:n=(lptr%)*4:srow%=0
1236 FOR i=0 TO 3:scol%=i*20:n=n+1:F_write
1237 END DEFine

1239 DEFine PROCedure F_write
1240 IF n>ftot% OR n<1:RETurn
1241 flen%=LEN(DFile$(n,1)):slen%=LEN(DDIR$)        flen% Filename length slen% SubDIRectory length
1242 IF flen%-slen%>18:flen%=18+slen%
1243 INK#1,fink%(n):CURSOR#1,8+scol%*6,srow%*10   fink% file ink print colour
1244 PRINT#1,DFile$(n,1,1+slen% TO flen%)&FILL$(' ',18+slen%-flen%)          Filename - Main Screen
1245 INK#0,5:CURSOR#0,px%,6:PRINT#0,DFile$(n,1):CLS#0,4:IF st%<>1:RETurn     Action Window
1246 INK#0,5:CURSOR#0,24,20:PRINT#0,'Select using ←↑↓→ ▬ Alt ↑↓ ↵ ':CLS#0,4
1247 BLOCK#0,12,3,130,24,5:BLOCK#0,2,4,198,22,5:CURSOR#0,300,6:PRINT#0,DFile$(n,2)  File Stats
1248 END DEFine    Use of BLOCK for Spacebar and Return Tail

1250 DEFine PROCedure F_clear                     Clear marked files
1251 FOR sc=1 TO ftot%:fink%(sc)=5                ftot% File Total sc set clear (ink colour)
1252 fs%=(lptr%*4)+1:fe%=(lptr%+16)*4:IF fe%>ftot%:fe%=ftot%
1253 FOR n=fs% TO fe%:Fscr_posn:F_write           fs% file start fe% file end
1254 END DEFine
```

```
1260 REMark QBITS FTidy File Mamgement

1261 DEFine PROCedure K_Chk                                  Note: INKEY$ Check
1262 REPeat chk_lp
1263   k=CODE(INKEY$(#0,-1))
1264   SELect ON k=78,110:chk=0:EXIT chk_lp                  Note: Answer 'n,N' no
1265   SELect ON k=89,121:chk=1:EXIT chk_lp                  Note: Answer 'y,Y' yes
1267   SELect ON k=10:IF cmd=9 OR cmd=10:chk=2:EXIT chk_lp  Note: Answer 'Enter' for VIEW & ZIP
1268 END REPeat chk_lp
1269 END DEFine

1271 DEFine PROCedure F_Copy                  Select Single/Multiple Files   COPY  win1_boot
1272 px%=96:mark%=7:nm%=stot%+1:st%=1:F_select:st%=0
1273 CURSOR#0,px%,6:PRINT#0,DDIR$;'File(s) (y/n)':CLS#0,4:fnum=n:K_Chk
1274 IF chk=1:cn%=0:ELSE nm%=1:F_clear:n=fnum:RETurn
1275 FOR n=stot%+1 TO ftot%                                  COPY  win1_QBITS_File(s) (y/n)
1276   IF fink%(n)=7
1277     cn%=cn%+1:CFile$(cn%,2)=DFile$(n,2)                 Note: Copy Stats
1278     CFile$(cn%,1)=DFile$(n,1,1+LEN(DDIR$) TO LEN(DFile$(n,1)))  Note: Copy Filename (less SubDIR)
1279   END IF
1280 END FOR n
1281 SD$=DD$:SDIR=DDIR$:TD$=DD$:TDIR$=DDIR$:F_Target
1282 END DEFine

1284 DEFine PROCedure F_Target        Select Target DIRectory   COPY QBITS_ File(s) TO win1_QBITS_
1285 REPeat tag_lp                                             Change (D)rive (S)ubDIR  (C)OPY File(s)
1286   CURSOR#0,24, 6:PRINT#0,'  COPY       Files(s)   TO  '&TD$&TDIR$:CLS#0,4
1287   CURSOR#0,24,20:PRINT#0,'Change (D)rive (S)ubDIR (C)OPY File(s)'
1288   k=CODE(INKEY$(#0,-1))
1289   SELect ON k
1290     =68,100:px%=138:SelDrv  :TD$=DD$                    :REMark D (Drive)
1291     =83,115:px%=170:SubDIR :TDIR$=DDIR$                 :REMark S (SubDIR)
1292     =67, 99:EXIT tag_lp                                 :REMark C (Copy) Exit loop
1293   END SELect
1294 END REPeat tag_lp
1295 IF SD$&SDIR$=TD$&TDIR$:nm%=1:F_clear:n=fnum:RETurn :ELSE CLS#0:F_Copy2
1296 END DEFine

1298 DEFine PROCedure F_Copy2                  COPY the selected file(s)
1299 FOR n2=1 TO cn%
1300   str$=CFile$(n2,1):chk=1                 Note: CFile$(n2,1)=DFi;e$(n2,1) less SDIR$
1301   CURSOR#0,24,6:PRINT#0,'  COPY   '&str$&' TO '&TD$&TDIR$:CLS#0,4
1302   FOR n1=stot%+1 TO ftot%
1303     IF str$==DFile$(n1,1,1+LEN(TDIR$) TO LEN(DFile$(n1,1)))   Note: Target with same Filename
1304       INK#0,3:CURSOR#0,6,20:PRINT#0,CFile$(n2,2)&'     '&DFile$(n1,2)
1305       INK#0,5:CURSOR#0,340,6:PRINT#0,' Overwrite y/n':K_Chk
1306       IF chk=0:NEXT n2
1307     END  IF                  Note: chk=1 Filename exist 'y=yes' Delete then Copy
1308     IF chk=1:DELETE TD$&TDIR$&str$:COPY SD$&SDIR$&str$ TO TD$&TDIR$&str$
1309   END FOR n1
1310 END FOR n2
1311 FileDIR                      Note: Display Updated Volume Info & Filenames of Target device
1312 END DEFine
```

```
1314 DEFine PROCedure F_Delete
1315 px%=96:mark%=7:nm%=stot%+1:st%=1:F_select:st%=0
1316 fnum=n:fdel%=ftot%
1317 FOR n=stot%+1 TO ftot%
1318  IF fink%(n)=7
1319   CURSOR#0,96,6:PRINT#0,DFile$(n,1)&' (y/n)':CLS#0,4:K_Chk
1320   IF chk=1:DELETE DD$&DFile$(n,1):fdel%=fdel%-1:fink%(n)=0:Fscr_posn:F_write
1321   IF chk=0:fink%(n)=5:Fscr_posn:F_write
1322  END IF
1323 END FOR n
1324 IF LEN(DDIR$)>0 AND fdel%=0:DELETE DD$&DDIR$:dl%=dl%-1:DDIR$=SubDIR$(dl%)
1325 IF fdel%<ftot%:FileDIR:ELSE nm%=1:F_clear:n=fnum
1326 END DEFine
```

```
DELETE win1_SpaceInvader_fnt
```

```
1328 DEFine PROCedure F_Run(act)                    Action Filename EXEC
or LRUN
1329 px%=96:mark%=5:nm%=stot%+1:st%=1:F_select:st%=0
1330 CURSOR#0,96,6:PRINT#0,DFile$(n,1)&' (y/n)':CLS#0,4:K_Chk
1332 IF chk=1 AND act=1:LRUN DD$&DFile$(n,1)
1331 IF chk=1 AND act=2:IF '_obj' INSTR DFile$>0:EXEC DD$&DFile$(n,1)
1333 fink%(n)=5:Fscr_posn:F_write
1334 END DEFine
```

```
LRUN   win1_QBITS_Darts_v3
EXEC   win1_progs_darts_v3_obj
```

Note: Select **Filename_obj** files with EXEC to run as a JOB. Exit from Program cancels JOB. LRUN command from an **_obj** file is not actionable. Use CTRL-C to switch back to the SuperBASIC Interpreter.

```
1336 DEFine PROCedure F_Rename
1337 px%=96:mark%=5:nm%=stot%+1:st%=1:F_select:st%=0
1338 CURSOR#0,96,6:PRINT#0,DFile$(n,1)&' (y/n)':CLS#0,4:K_Chk
1339 IF chk=0:fink%(n)=5:Fscr_posn:F_write:RETurn          IF NO Clear Highlight &
RETurn
1340 INK#0,5:CURSOR#0,px%,6:PRINT#0,DDIR$:CLS#0,4
1341 str$=DFile$(n,1,1+LEN(DDIR$) TO LEN(DFile$(n,1)))
1342 sl%=LEN(str$):cp%=sl%+1:sm%=36-LEN(DDIR$):px%=px%+LEN(DDIR$)*6:Ln_Ed      Edit
Filename
1343 IF str$='':str$=DFile$(n%,1):fink%=5:Fscr_posn:F_write:n=fnun:RETurn
1344 FOR n1=1 TO ftot%:
1345    IF str$==DFile$(n1,1)
1346       INK#0,5:CURSOR#0,24,20:PRINT#0,'Filename Exists':CLS#0,4          Note: Check if Rename
Exists
1347       PAUSE 50:fink%(n)=5:Fscr_posn:F_write:RETurn
1348    END IF
1349 END FOR n1
1350 COPY DD$&DFile$(n,1) TO DD$&str$:DELETE DD$&DFile$(n,1):FileDIR
1351 END DEFine
```

```
RENAME win1_QBITS_Darts_v3
```

Note: A change of Filename uses Copy and then Delete and does not check the available storage on target device. Please take note if a Large File is being renamed this might lead to the action being rejected and possible loss of Data!.

```
1353 DEFine PROCedure F_View
1354 px%=96:mark%=5:nm%=stot%+1:st%=1:F_select:st%=0
1355 CURSOR#0,96,6:PRINT#0,DFile$(n,1)&' (y/n Enter)':CLS#0,4
1356 fnum=0:K_Chk:IF chk=0:fink%(n)=5:Fscr_posn:F_write:RETurn
1357 CURSOR#0,240,20:PRINT#0,'<SPACEBAR> to continue... <ENTER> to Exit'
1358 CURSOR#0,160,20:PRINT#0,'Bytes: ':CLS#1
1359 OPEN_IN#9,DD$&DFile$(n,1):char%=0:fline%=0:row%=0:fbyts=0
1360 REPeat View_lp
1361   k$=INKEY$(#9,-1):IF EOF(#9):CLOSE#9:K_Chk:IF chk=2:EXIT View_lp
1362   IF chk=1:PRINT#1,k$;                                    ASCII Printout
1363   IF chk=2:CURSOR#1,6+char%*15,row%:PRINT#1,HEX$(CODE(k$),8)   HEX Printout
1364   char%=char%+1:fbyts=fbyts+1:CURSOR#0,200,20:PRINT#0,fbyts
1365   IF chk=1 AND char%>=74 OR chk=1 AND k$=CHR$(10)          ASCII New Line
1366     char%=0:fline%=fline%+1
1367     IF fline% MOD 16=0:IF INKEY$(-1)=CHR$(10):CLOSE#9:EXIT View_lp   ASCII New Page
1368   END IF
1369   IF chk=2 AND char% MOD 32=0                              HEX New Line
1370     char%=0:fline%=fline%+1:row%=row%+10
1371     IF fline% MOD 16=0:IF INKEY$(-1)=CHR$(10):CLOSE#9:EXIT View_lp   HEX New Page
1372     IF row%>150:row%=150:SCROLL -10
1373  END IF
1374 END REPeat View_lp
1375 nm%=1:CLS#1:F_clear:n=fnum
1376 END DEFine
```

VIEW win1_QBITS_Darts_v3

```
1378 DEFine PROCedure F_ZIP
1379 px%=96:mark%=5:nm%=stot%+1:st%=1:F_select:st%=0:ALTKEY 'z'
1380 INK#0,7:CURSOR#0, 24,6:PRINT#0,'COMPILE ';DFile$(n,1);' (y/n Enter)':CLS#0,4
1381 K_Chk:IF chk=0:RETurn :END IF :IF chk=2:RTme=1:ELSE RTme=0
1382 CLS:WINDOW 364,138,72+gx,54+gy:PAPER 208:CLS:BORDER 1,7
1383 CSIZE 2,1:CURSOR 132,20:PRINT 'COMPILER':CSIZE 0,0
1384 CLS#0:CURSOR#0,66,6:QLIB_USE:IF eck=1:Load_Qlib(RTme):eck=0:PAUSE 60
1385 CLS#0:CURSOR#0,66,6:cl%=INT(LEN(DFile$(n,1))/2)
1386 CURSOR 24, 60:PRINT FILL$(' ',18-cl%)&'LIBERATE ';DD$&DFile$(n,1);',';
1387 CURSOR 92, 80:PRINT 'Press ALT-z to LOAD & COMPILE'
1388 CURSOR 56,120:PRINT 'Then CTRL-SPACE & ALT-f for QBITS_FTidySE'
1389 ALTKEY 'z','LOAD '&DD$&DFile$(n,1);'LIBERATE '&DD$&DFile$(n,1)&',';CHR$(10)
1390 STOP
1391 END DEFine
```

ZIP win1_QBITS_QBITS_Darts_v3

Note: To COMPILE Filename_bas (y/n Enter) – Enter will Compile with Qlib_Runtime. The QL Platform requires O/S SMSQ/E and Sinclair QL Forum Edition 2020 of QLIBERATOR for QPC2.

```
1393 DEFine PROCedure Load_Qlib(RTme)
1394 add1=RESPR(15064):LBYTES dev$&'Qlib_sys',add1:CALL add1
1395 add2=RESPR(49004):LBYTES dev$&'Qlib_obj',add2:CALL add2
1396 REMark Q_LIBERATOR  Settings
1397 IF RTme=0:QLIB_USE dev$,dev$,72+gx,54+gy,"0011010100" :REMark RunTime Off
1398 IF RTme=1:QLIB_USE dev$,dev$,72+gx,54+gy,"0011110100" :REMark RunTime On
1399 END DEFine
```

Note: QLIB_USE attributes : Load_Device for QLIB_OBJ, QLIB_HELP, Window x,y coordinates, "Option Bits" 1 to 8 - STATS-DEBUG-LINES-NAMES-RUN-AUTO-BEEP-WINDS - [9&10 Reserved].

```
1401 REMark QBITS FTidy Line Editor
```
Note: The Line Editor restricts characters to numeric 0 to 9 [ASCII 48-57], the UPPER/lower-case Alphabet A-z [ASCII 65-90 & 97-122] plus underscore '_' [95]. Position the character highlight (Underline **Ln_Cur**) with Left Right Cursors, then Add (**Add_Chr**) a new or Delete (**Del_Chr**) existing Character.

```
1403 DEFine PROCedure Ln_Ed
1404 INK#0,5:CURSOR#0,24,20
1405 PRINT#0,'Edit ← → BkSp (← CTL→) Del ↵ Rtn':BLOCK#0,2,4,198,22,5
1406 REPeat Ed_lp
1407   Ln_Prn:Ln_Cur:k$=INKEY$(#0,-1):k=CODE(k$)
1408   SELect ON k
1409     = 10:Str_chk:EXIT Ed_lp
1410     = 48 TO 57, 65 TO 90,95, 97 TO 122:Ln_Prn:Add_chr          ASCII codes available for Filenames
1411     =194:IF cp%>1:cp%=cp%-1:Del_chr                            cp% cursor position
1412     =202:Del_chr
1413     =192:IF cp%>1:cp%=cp%-1
1414     =200:IF cp%<sl%+1:cp%=cp%+1                                sl%  string length
1415   END SELect
1416 END REPeat Ed_lp
1417 END DEFine
```


```
1419 DEFine PROCedure Str_chk
1420 REPeat str_lp
1421   IF '_' INSTR str$(LEN(str$))=1:cp%=sl%:Del_chr:Ln_Prn      Note: Removes any EOL '_'
1422   IF '_' INSTR str$(LEN(str$))=0:PAUSE 30:EXIT str_lp
1423 END REPeat str_lp
1424 END DEFine
```


```
1426 DEFine PROCedure Ln_Prn
1427 IF LEN(str$)>sm%:str$=str$(1 TO sm%):cp%=sm%              Note: Truncate to sm% max string
length
1428 INK#0,7:CURSOR#0,px%,6:PRINT#0,str$:CLS#0,4
1429 END DEFine

1431 DEFine PROCedure Ln_Cur
1432 BLOCK#0,8,1,px%+cp%*6-6,15,2                              Note: px% x start position
1433 END DEFine

1435 DEFine PROCedure Add_chr
1436 IF cp%  =1 AND sl%=0     :str$=str$&k$                          add to string
1436 IF cp%>=1 AND cp%<sl%:str$=str$(1 TO cp%-1)&k$&str$(cp% TO sl%)  add in string
1438 IF cp%>=1 AND cp%=sl%:str$=str$(1 TO cp%-1)&k$&str$(cp%)        add one before end
1439 IF cp%>  1 AND cp%>sl%:str$=str$&k$                            add to end of string
1440 IF cp%=sm%:str$(cp%)=k$                                        change last character
1441 IF sl%  <sm%:sl%=sl%+1  :ELSE sl%=sm%                  sl% string length sm% max length
1442 IF cp%<sm%:cp%=cp%+1:ELSE cp%=sm%                      cp% character position
1443 END DEFine

1445 DEFine PROCedure Del_chr
1446 IF cp%=sl%:str$=str$(1 TO sl%-1):sl%=sl%-1                                  delete end of string
1447 IF cp%>=1 AND cp%<sl%:str$=str$(1 TO cp%-1)&str$(cp%+1 TO sl%):sl%=sl%-1    delete in string
1448 IF cp%=sm%:str$=str$(1 TO sm%-1):cp%=cp%-1:sl%=sm%-1                       delete last character
1449 IF cp%=1 AND sl%=1:str$="":sl%=0                                           Null string
1450 END DEFine
```

```
1452 REMark QBITS FTidy Graphics

1454 DEFine PROCedure KInfo(ch,col,x,y,r)
1455 INK#ch,col:CIRCLE#ch,x,y,r:LINE#ch,x,y-r/1.5 TO x,y:POINT#ch,x,y+.5
1456 END DEFine

1458 DEFine PROCedure KQuit(ch,col,x,y)
1459 INK#ch,col:CIRCLE#ch,x,y,r:LINE#ch,x,y+r*1.5 TO x,y-.1
1460 END DEFine

1462 DEFine PROCedure GDrive(col,x,y)
1463 FILL#2,1:INK#2,col
1464 LINE#2,x-4,y TO x,y+2 TO x+4,y+1 TO x+4,y-1 TO x,y-3.5 TO x-4,y-2 TO x-4,y
1465 FILL#2,0:INK#2,0
1466 LINE#2,x-4,y TO x,y-1 TO x+4,y+1:LINE#2,x,y-3.5 TO x,y-1
1467 LINE#2,x-3.6,y-1.5 TO x-.5,y-2.6:INK#2,7
1468 END DEFine

1470 DEFine PROCedure GFolder(col,x,y)
1471 FILL#2,1:INK#2,col
1472 LINE#2,x-3,y+2 TO x-2.6,y+2.4 TO x-1,y+2.4 TO x,y+2 TO x+2,y+2
1473 LINE#2 TO x+2,y+1 TO x+3,y+1 TO x+2,y-1.8 TO x-3,y-1.8 TO x-3,y+2
1474 FILL#2,0:INK#2,0
1475 LINE#2,x-3,y-1.8 TO x-2,y+1 TO x+4,y+1:INK#2,7
1476 END DEFine
```

Note: As an exercise the Graphics below were seen as possible symbols for use with the Pointer Environment.

```
2000 REMark QBITS Pointer Graphics

2002 DEFine PROCedure KExit)ch,co,x,y)
2003 INK#ch,col:LINE~ch,x+1.8,y+2 TO x-1,y+2 TO x-1,y-2 TO x+2,y-2
2004 LINE#ch,x,y TO x+3,y:LINE#ch,x+2,y+1 TO x+3,y TO x+2,y-1 TO c+2,y+1
2005 END DEFine

2007 DEFine PROCedure GDisk(ch,col,x,y)
2009 FILL#ch,1:INK#ch,col:LINE#ch,x-2,y+2 TO x+1,y+2 TO x+2,y+1
2010 LINE#ch TO x+2,y-2 TO x-2,y-2 TO x-2,y+2:FILL#ch,0
2011 END DEFine

2012 DEFine PROCedure GSave(ch,col,x,y)
2008 GDisk ch,col,x,y:INK#ch,0:LINE#ch,x-1,y TO x+1,y
2009 LINE#ch,x-1,y TO x+1.5,y:LINE#ch,x-1,y-1 TO x+1.5,y-1
2010 END DEFine

2012 DEFine PROCedure GLoad(ch,col,x,y)
2013 GDisk ch,col,x,y:INK#ch,0:LINE#ch,x-1.6,y+.5 TO x+1.2,y+.5
2014 LINE#ch TO x-1.6,y-1.6 TO x+1.2,y-1.6 TO x-1.6,y+.5
2015 END DEFine

2017 DEFine PROCedure GCopy(ch,col,x,y)
2018 GDisk ch,col,x,y:INK#ch,0:LINE#ch,x-.5,y-.5 TO x+1.8,y-.5
2019 LINE#ch,x-1.5,y+1 TO x-1.5,y-1.5 TO x+1.8,y-1.5
2020 END DEFine

2022 DEFine PROCedure GDelete(ch,col,x,y)
2023 GDisk ch,col,x,y:INK#ch,0:LINE#ch,x-.4,y+1 TO x+.6,y+1
2024 LINE#ch,x-1,y+.5 TO x+1,y+.5 TO x+1,y-1.5 TO x-1,y-1.5
2025 LINE#ch TO x-1,y+.5:LINE#ch,x,y-1.5 TO x,y+.5
2026 END DEFine

2028 DEFine PROCedure GRename(ch,col,x,y)
2029 GDisk ch,col,x,y:INK#ch,0:LINE#ch,x,y+.5 TO x,y-1.5
2030 LINE#ch,x-1.2,y+.5 TO x+1.2,y+.5:LINE#ch,x-1.2,y-1.5 TO x+1.2,y-1.5
2031 END DEFine
```

Floppy Disk