Contents

Introduction

Welcome to QD - thank you very much for purchasing this "swiss army knife" of software. You will find that QD contains so many useful features that it may take a while until you get used to them. Please read the manual at least once - you will get an idea what it can do for you. Use it as your "standard" text editor to get used to the basic editing features. Then, if you want it to do specific jobs, use the manual again. The possibility of passing parameter strings to QD will make it very, very versatile for lots of different tasks.

The presence of the Pointer Environment, Window Manager, Hotkey System II and the Menu Extension is essential for QD. We will not discuss the various extensions in the QD manual, a separate manual for the Pointer Environment, Window Manager and Hotkey System II is available from Jochen Merz Software. If you do not have any knowledge about it, just boot the QD floppy disk and QD will work automatically. Have a look at the BOOT file on the disk if you want to integrate it into your existing BOOT file.

Pointer Environment (file PTR_GEN). The first resident extension required by QD which deals with mouse and adds "real", non-desctructive windows to your QL system. If you own SMSQ/E, don't load it, it is built into SMSQ/E.

Window Manager (file WMAN). The second resident extension required by QD which deals with menus and objects inside windows. If you own SMSQ/E, don't load it, it is built into SMSQ/E.

Hotkey System II (file HOT_REXT). The third resident extension required by QD. If you own SMSQ/E, don't load it, it is built into SMSQ/E.

Menu Extension (file MENU_REXT). The fourth resident extension required by QD. It provides you with pre-defined menus, like the file-select window. This extension is not (yet) built into SMSQ/E, therefore you always need to load it.

Naturally, all rights to all the files supplied are the property of the authors, except PTR_GEN, WMAN, HOT_REXT and CONFIG which are licensed by QJump. The same applies to this manual. We are not liable for any error within this manual or the files.

You can get free updates of QD at any time. Send us the QD master medium and a S.A.E. together with two international reply coupons, then you will get the latest version of QD. This applies to updates only and not to major revisions of the software. In case the medium has become faulty in use, please enclose an additional reply coupon for a replacement.

You have noticed that these days there are not many people producing software for the QL. Good software has become a rare phenomenon. We aim to supply only the best software to QL users.

None of the program files is copy-protected. We follow this policy to make your life as easy as possible. Therefore we ask you not to give copies of the program away. You can imagine the amount of work required to produce a program like QD, which has to be profitable for us. If this does not happen, there are certain consequences: Higher prices, a really annoying copy protection scheme or the end of developement on this product (no updates, for example).

This should not happen, so please do not lend or give any copies of this program to friends. Encourage them instead to buy their own copy.

Our problem is that this request is aimed at the fair-minde users, as software-pirates do not have an original manual. So please, support us and all the honest users - including, not to forget, yourself! Thank you very much.

First you have to make yourself a backup of the original QD medium. You can do it as shown in the User Guide by using COPY, or, if you have SuperToolkit II, by WCOPY. QRAM or QPAC II owners can do it even more easily with the Files menu.

Credits: Many thanks to Tony Tebby who helped us very much with his ideas, help and support. Without his help, modifications in the Pointer Interface and Window Manager QD would never look like it does.. Thanks also to Bernd Reinhard and Oliver Fink for their ideas, support and for Oliver's QD5 Assembler Thing. Thanks also to all the users giving me ideas how to improve QD.

The QD Concept

QD has been designed to give you a fast **and** user-friendly editor. Note that QD is not meant to be a word processor but an ASCII text editor. QD is not a replacement for Quill or text87, for example, because it is a totally different kind of program. It is also not a direct competitor to The Editor.

As stated before, QD is user-friendly. People who know QRAM or QPAC II will soon become familiar with QD. This was the aim of the pointer environment. We threw out all unnecessary things to get the highest possible editing speed.

We hope that we chose the best compromise. If you feel any useful features are missing, please let us know.

QD was aimed at the large community of QL programmers who need a good text editor. These are, to name a few, programmers using assembly language, C, Pascal, BCPL and Fortran. SuperBASIC programmers also benefit from QD, since powerful commands such as search/replace are not available in the Toolkit II SuperBASIC editor.

For all these people right justification or similar features are not necessary. Programmers require fast operations such as immediate block moves. They have no time to wait for loading and saving of text. Memory should be reserved dynamically and there should be no need for garbage collection which takes a lot of time. A programming editor should not be file-orientated as this is not the fastest way to manage a text and you cannot remove the medium while editing the text.

So we tried to get the best. We made a product which took all these points into account: memory is allocated only when it is needed; there is no garbage collection; and it is not file-orientated. You have to access files only when you do load and save operations.

We have designed the best display and cursor-handling we could think of. Most editors only allow editing of characters, trailing spaces are not allowed. If you move the cursor beyond the last character of the line, the cursor skips to the start of the next line. This makes editing more difficult and unnecessary spaces are saved to the file. QD lets you move the cursor anywhere on the screen, spaces beyond the last character of a line will be removed when the line scrolls out of the window.

Starting QD

If everything has so far been done as instructed and all resident extensions have been installed, you can start QD for real work:

```
EXEP flp1_QD
```

After a few seconds loading time, QD shows its window (if there is enough memory).

You may pass a filename to QD which makes QD loading itself first and then automatically loads in the given file:

```
EXEP flp1_QD;"filename"
```

You can pass additional information in the parameter string, but this is described in another chapter of this manual.

You can make QD resident and execute many versions of QD with only having the code one time in memory, as QD is re-entrant. It is loaded as a thing - with or without additional filename-parameter. You already know how it goes, but to save your time here is a short example:

```
ERT HOT_CHP ('d',flp1_QD)                load QD resident
ERT HOT_WAKE('b',flp1_QD;'flp1_boot')    to start QD with BOOT
EXEP QD                                  start QD
```
or press [ALT] [d] to do the same

Again, to remind you, press [F1] or select HELP to get help! Note that the help file (normally flp1_QD_HELP) must be present. You can change the filename where QD is looking for the help file with CONFIG or, during run-time in the F5 STATUS menu.

Cursor movement

Within the text-area (this is the large window area) the pointer looks like a normal text cursor. If you leave the text-area or pull down a window it changes to an arrow. Only when the pointer is in the text-area, can you use the following cursor-control codes to move the cursor and/or delete something.

LEFT/RIGHT/UP/DOWN	moves the cursor one character in the selected direction. If it leaves the text-area, the window will be panned or scrolled to hold the cursor within the text-area.
ALT LEFT	moves the cursor to the start of the line. If it is not visible, the window will be panned to make it visible.
ALT RIGHT	moves the cursor to the end of the line. If it is not visible, the window will be panned to make it visible.
ALT UP	scrolls text upwards.
ALT DOWN	scrolls text downwards.
SHIFT LEFT	moves the cursor one word left. If the start of the word is not within the visible part of the text-area, the text will be panned.
SHIFT RIGHT	moves the cursor one word right. If the start of the word is not within the visible part of the text-area, the text will be panned.
SHIFT UP	moves the cursor one page up.
SHIFT DOWN	moves the cursor one page down.
SHIFT ALT LEFT	pans text left, the cursor holds its position within the text.
SHIFT ALT RIGHT	pans right, the cursor holds its position within the text.
SHIFT ALT UP	moves the cursor to the start of the text. This is a much easier way of doing a GOTO.
SHIFT ALT DOWN	moves the cursor to the end of the text.

CTRL LEFT	deletes one character left. If the first character of the line has been deleted and the cursor is at the leftmost position, then the current line will be appended to the line above. In fact you remove the ENTER (which you never see) at the end of the line before. If the length of both lines together is more than the maximum line length, a warning beep will sound.
CTRL RIGHT	deletes the character under the cursor.
CTRL DOWN	deletes the whole cursor line. All following lines move one line up.
ALT CTRL LEFT	deletes the all characters to the left of the cursor in the current line.
ALT CTRL RIGHT	deletes all characters in the cursor line to the right of the cursor.
SHIFT CTRL LEFT	deletes word to the left of the cursor.
SHIFT CTRL RIGHT	deletes word to the right of the cursor.
ALT CTRL DOWN	inserts a blank line directly under the line which holds the cursor without changing the position of the cursor.
ALT CTRL UP	undoes all the changes to the cursor line which took
UNDO	place after the line was scrolled into the window. The UNDO-key only exists on the ATARI-SMSQ/E.
ENTER	moves the cursor to the next line and inserts an ENTER. If the cursor is within text, the left part of the line will be put into the next newly created line.
SHIFT ENTER	insert a Carriage Return CHR$(13). This function will only be used if the text edited is an Archive- or TSL-program file or it is necessary that every line has a Linefeed and Carriage Return at its end.
SHIFT SPACE	toggles between insert and overwrite mode.

The mouse buttons have some additional functions in the edit-window. These functions are not identical to [SPACE] or [ENTER], this time! The left mouse button usually marks a block (see BLOCK). The right mouse button pops up the context menu (see CONTEXT menu).

There are two special cursor control combinations, which work dependent on the current character under the cursor. We call them bracket moves. The following characters are accepted as pairs of brackets:

( ) { } [ ] < > «»

In addition, both string characters " and ' are treated as legal pairs of the same suite.

When the cursor is over any of the character above, and one of the following key-combinations is pressed, then the cursor jumps to the matching bracket if possible. All bracket pairs which lie between those matching brackets are taken into account (i.e. the bracket level is checked) and they are skipped therefore. If there is no matching bracket, the cursor stays where it is.

ALT SHIFT CTRL RIGHT	searches the closing bracket which matches the open bracket under the cursor. If it does not exist, the cursor stays where it is. If the character under the cursor is not an open bracket, nothing happens.
ALT SHIFT CTRL LEFT	searches the open bracket which matches the closing bracket under the cursor. If it does not exist, the cursor stays where it is. If the character under the cursor is not a closing bracket, nothing happens.

The start: load a file

But enough theory, let's type something. Move the cursor to any point in the text-window and start! You can move the cursor during typing at any time with the mouse or the cursor keys, of course. Remember, you can always press [F1] or select HELP for help.

Now, we can edit an existing text file for example. Here is the way to do it:

Let us assume you started QD without passing a filename. Normally you see the name of the current file in the top right part of the window. You will see, your text has no name. It will get a name as soon as you save it.

Select FILES (you already know pressing [F2] is faster than the mouse!), then select LOAD TEXT. If you have already edited a text with QD, another window comes up and you are asked whether you want to forget the current text or not. A word on this YES/NO request: you can always press [ESC], this counts as NO!

Now you may enter the name of the file you wish to load. You can do it in a very comfortable way - using the Menu Extension: another large window will appear. Select the item under the filename prompt and enter the filename. There are many ways of doing it; they are all explained in the part of this manual which deals with the FILE SELECT window.

If you are using SuperToolkit II or a disc controller, you may enter the filename only, without a drive name. The program will then use the DATA_USE default drive. At its first attempt, the program uses the entered name to open the file, but if this fails, the DATA_USE default drive will be placed before the name and another attempt will be made. For example, if DATA_USE is flp2_ and you wish to load flp2_UPDATES_TEXT, you just have to enter UPDATES_TEXT. You can configure this behaviour and use a "local" default directory, which is valid only or the current QD.

In case there is an error another window will pop up and the error will be reported. Further action depends on the type of the error.

The Parameter String

You can pass a parameter string when you start QD which contains much more information. Parameters given in the string overwrite configured settings.

All parameters are separated by spaces. You may miss any parameter from the end. The order is:

Name of the file which should automatically loaded when QD is started.
Name of the local directory for QD.
Filename extension.
Name of the F10-Thing.
Name of the file containing the help text.
Tabulate value.
Left margin.
Right margin.
Job name.

Examples:

```
EXEP QD;"win1_basic_demo_bas win1_basic_ _bas"
```

starts a QD, loads the file win1_basic_demo_bas and sets directory and extension.

```
EXEP QD;"flp1_BOOT"
```

starts a QD editing flp1_BOOT.

You may also set the parameters in a different order by placing a key in front of it. A key is introduced by a '\', followed by a letter (upper- or lower case). The keys are:

`\F`	filename, followed by the name
`\D`	directory, followed by the directory
`\E`	extension, followed by the extension
`\T`	thing, followed by thing name
`\H`	name fo the file containing the help text
`\A`	tabulate value
`\L`	left margin
`\R`	right margin
`\Q`	quit immediately. No parameters.
`\P`	text is protected and cannot be modified. No parameter.
`\J`	job-name.
`\S`	search string. If QD is executed with a file, then this file is automatically searched for the string given here.
`\ON`	activates tab-compression for all eight extensions.
`\OFF`	deactivates tab-compression for all eight extensions.
`\+`	followed by preamble-string
`\-`	followed by postamble-string
`\I`	inserts the current contents of the scrap right at the start of the text. This key can be combined with loading a file. In this case, the file is loaded first and the scrap is inserted above the file.

`\M0`	don't match case on search
`\M1`	match case on search
`\>`	followed by filename specifies new filename for save This is very useful if you wish to create texts (e.g. faxes) out of a template file, but you do not want to overwrite the template file by mistake. Example: `EXEP QD;'win1_txt_template_txt \>win1_fax_new_txt'` loads the file template_txt automatically into QD but saves to fax_new_txt.
`|`	Activates the wordwrap icon.
`\C`	followed by a number defines the initial width of QD's window in characters. It is rounded to a multiple of 20.
`\U`	followed by a character defines the usage (see below, F5 - Usage). If no character is given, usage is "general". A is Assembler B is BASIC M is Message U is user-defined.

More examples for a better understanding:

```
EXEP QD;'\E _asm \j Edit'
```

starts QD and sets the extension, renames QD to Edit.

```
EXEP QD;'\T Parser \D win1_basic_ \F win1_demo'
```

starts QD with win1_demo, sets the directory to WIN1_BASIC_ and enables the F10-Thing to call a Parser.

```
EXEP QD;'win1_fred_asm \OFF \s HOT_'
```

starts QD with the file win1_fred_asm, does not decompress it under any circumstance, and searches for the string "HOT_". If found, the cursor is positioned over it, otherwise the cursor is placed at the start of the text.

General information

First of all some information on YES/NO requests. [ESC] is always the same as NO. If you deselected confirmation request, you will not get confirmation requests. But if you selected it, windows will come up following a clever scheme: all request which would modify or delete a file will always come up. Request which only delete or modify the current text do not appear if the text has been loaded and no modification was done.

You surely noticed already that sometimes a few menu items are unreadable. You also cannot select them. QD handles its menus in an intelligent way: all functions which do not make sense at the time you pull down the menu are not available. To give some examples: block-operations are not very useful as long as no text is marked. If there is no file, there is no need to save it. Without having loaded a file, a GOTO doesn't make sense either.

The current version of QD allows you to load in files containing lines of any length. If a line is longer than the maximum line length defined with CONFIG, the line will be continued onto the next screen line, starting with a '⟵'-character. If this file is to be saved, the marker will be removed and one long line created. Note that the saved file may have a different length (even if you made no modification), as QD optimises SPACEs.

At the top left corner of the QD window you see the move window item and the change window size item. If you select one of these, the pointer will change its shape. Move it and the window will be moved or resized, depending on the item chosen. If the window fills the screen completely, then moving does not make any sense, of course. If you press [ENTER] or the right mouse button over the resize symbol, then the window will be resized to the maximum size possible. The next time you press [ENTER] or the right mouse button on this item, the window goes back to its previous origin with its previous size.

Next to most numerical entry fields you will find two small arrows. They allow you to modify the number without actually editing it. When you left-click an arrow once, you will increase/decrease the value by one. If you right-click an arrow, the step will be larger (depending on the usage of the value). Tab's will go in steps of 8, for example, others will usually step by 10 or 100.

The Scroll-Bar

The scroll-bar on the right side of the edit window contains a lot of different functions: it shows the textsize, the position of the current window in the whole text and it may be used for fast scrolling and splitting. Imagine the whole bar is the total length of the text. The block inside the bar is the text you currently see in the window. The size of the bar is relative to the whole size of the text, i.e. the smaller the bar the larger is the text. The position of the block is also directly coupled to the position of the text. If the block is in the middle of the bar, the window contains lines of the middle of the text and so on.

You may, if you own a mouse, point to the bar and press the left mouse button. The window will scroll to this position (relative to the text size, of course). If you keep the button down and move the mouse vertically over the bar, the text will move corresponding to the bar. This is called 'dragging' and allows you to scroll very fast through the whole text. If you do not own a mouse, you will find it quite difficult to get the cursor out of the text window over the bar. Press [SHIFT] [CTRL] [ESC], and the cursor jumpts to the bar. Press it again, and you are back in the edit window.

The bar contains an additional feature: point to the bar again and press the right mouse button. You will notice that the bar splits at the pointer position into two sections. You now have two independent windows. You may control and edit the upper section in the same ways as always. The lower section is for viewing only, but may also be scrolled with its scroll bar. How do you join the two section? You already guessed it: just press the right mouse button again while pointing to the split.

Do not wonder that even an empty QD does not show a full-height block in the bar. QD always reserves more lines work-space, which means, the bar size always signals a minimum of the empty window size in lines.

Pick and Buttons

When you leave the QD window (not quit the program, just change over to another program which currently multitasks in your machine) there are many ways to get back into QD. The best known is [CTRL] [C]; you have to press it once or a few times, but eventually you will find the QD window. It has the disadvantage that, in common with HOT_PICK, the cursor appears in the middle of the window. If you sent a WAKE event together with the PICK, the cursor will come up at the position it was when you left QD. You can send a WAKE event by picking the window with HOT_WAKE or by pressing the right mouse button to pick it.

You may also put the current QD window to a button. A button is a small window which stands for another job. A QD button contains the text 'QD filename'. You can move it by pressing [SPACE], you can wake the button (i.e. make it come up in its previous full size) by pressing [ENTER] while the pointer is in the button window. The button will appear at the current pointer position or it will find its position in the button frame if you own QPAC II (then you can't move it). Press [ESC] when you are not in a sub-menu, select the Zzz-symbol to get QD in a button.

If QD is buttonised, the button will display the flashing "text changed" icon if the text in QD has been changed but not saved yet. This should remind you not to turn your system off, 'cause the changes will be lost. You can also remove the button and QD by pressing [ESC] while the pointer is over the button. If the text was changed but not saved then QD will not be terminated but woken instead.

The Toolbar

Just above the edit-window you can see a number of small pictures, we call them icons. If you do not own a mouse and you find that you cannot move the cursor/pointer out of the edit window, press [ALT] [TAB] and the pointer will jump over the question-mark-icon. You can now freely move it. If ALT TAB does not work, make sure, you have not defined an ALTKEY onto TAB, as this has a higher priority. The HOTKEY System will treat ALT TAB and it will never reach QD. The symbols from left to right:

Info (i-icon) gives you information about the version of QD.

Help (question-mark-icon) activates help or the help system. [F1] does the same.

Load (left disk icon) immediately calls the file-select window. If this icon was selected with the left mouse button or [SPACE], then the selected file will be loaded. If it was selected with the right mouse button or [ENTER] or [RETURN], then the selected file will be inserted.

Save (right disk icon) saves the current file. Old copies will be overwritten, without confirmation (even if you set confirmation on). If you selected this icon with the right mouse button or [ENTER] or [RETURN], then you may give it a new name before saving.

Print (printer-icon) pops up the print menu if you left-click this icon. If you right-click it or press [ENTER] or [RETURN], then it will print with the current settings of the print menu..

Search (magnifier-icon) pops up the search menu if you left-click it. A right-click will bring up the replace menu.

Continue search (magnify-icon with dots) repeats the last search/replace action. [F9] does the same.

Block-start (block with arrow left) turns the cursor into block-start mode. For more details look at F4-Blocks. [F7] does the same.

Block-end (block with arrow right) turns the cursor into block-end mode. For more details look at F4-Blocks. [F8] does the same.

Block move (block on waggon) turns the cursor into block-move mode, if left-clicked. For more details look at F4-Blocks. If you click the move block icon with the right mouse button, then you can insert the scrap. The cursor changes to the insert scrap shape, as it does when you select F2 - Insert scrap.

Block copy (block with camera) turns the cursor into block-copy-mode, if left-clicked. For more details look at F4-Blocks. If you click the copy block icon with the right mouse button, then you get into the "repeated copy" mode. You can copy the block as many times as you want. Leave this mode by pressing [ESC].

Block delete (block into trashbin) deletes the current block. In case the block is fairly large and confirmation request is selected, you will be asked for confirmation before the block is deleted.

Block into scrap (arrow pointing to scrap) will either overwrite the contents of the scrap with the currently selected block, if this icon was hit with the left mouse button or SPACE, or the block is added to the scrap if this icon was hit with the right mouse button or RETURN or ENTER.

Wordwrap (text with a red vertical bar) will enable or disable word wrap (see word wrap).

There may be other symbols further right, but they are dependent on Extension Things. If a QD Extension Thing adds sysmbols, it will explain them in the appropriate manual.
If no QD Thing is active or if one is active but does not add any icons, then five additional icons are visible:

Goto assembler label (arrow pointing to LABEL) is the same as F3 - Goto - Assembler-Label. Shortcut keystroke is [CTRL] [F6].

Goto BASIC procedure (arrow pointing to PROC) is the same as F3 - Goto - BASIC Procedure. Shortcut keystroke is [CTRL] [F7].

Goto BASIC funciton (arrow pointing to FN) is the same as F3 - Goto - BASIC Function. Shortcut keystroke is [CTRL] [F8].

Compress and expand (vice-symbol) will expand all control codes when left-clicked (same as F3 - Delete controlcodes) and, when right-clicked, pop up the tab compression menu.

Usage (text with red highlights) will refresh the text using the current usage setting, if left-clicked. When right-clicked, the usage menu is brought up. [CTRL] [F10] is the same as left-click.

The remaining items

You may toggle the insert/overwrite status by pressing [SHIFT] [SPACE] (on PC style keyboards, this is the "INS" key - how useful!). If "Insert" is selected, all entered characters will be inserted at the current cursor position. This is the 'normal' mode which you surely will know if you use QUILL or the Super-BASIC command line editor. If this item is "Overwrite", all characters will be overwritten with the characters entered.

Near the right top corner, you find the current line and column display. This little window is updated by a second job, to make sure that the main QD is not slowed down or disturbed at all. Both lines and columns count from 1.

If you start doing modifications to the text, a small "save to disk" symbol will start flashing right to the row/column display. This indicates that the current state of the text has not been saved, reminding you not to remove this QD unless you saved it. After saving, it disappears until the next modification happens.

Item Explanation

You can get explanatory help about menu items in the main menu (including the icons). Move the pointer over the desired item and wait a few seconds. A small text will pop up, explaining the function of this item. If it has two different functions for the two different mouse buttons, it will explain both functions (left button first, then right button). Of course, [SPACE] is the same as the left mouse button, [ENTER] or [RETURN] the same as the right mouse button.

The small window disappears as soon as you move the mouse or press any key.

You can disable this feature by typing the following line into BASIC before you start QD:

```
SET_DEFAULT 97,"-1"
```

Any other value instead of "-1" sets the delay. "50" is one second. Please note that the quotes are mandatory!

You can also configure MENU_REXT to make the change permanent. Use the program Menuconfig (see CONFIG section further on in the manual) but configure MENU_REXT instead of QD!

Wordwrap

It is now possible to make QD to wrap words automatically when the cursor reaches the right margin. We're talking about wordwrap only, not automatic re-format. When QD was designed many years ago, it was never planned to turn it into a wordprocessor. The internal structures are defined so that text editing is very comfortable: you can position the cursor anywhere you want, to the left of a left margin and to the right of a right margin etc. (you can't do this in wordprocessors like Quill or text87), but there is no concept of "paragraphs" - essential for automatic reformatting. But, wordwrap on its own is quite useful for writing short texts.

There is an icon in the icon bar which controls wordwrap. It is right to the "block into scrap" icon, nr. 14 if you start counting from left. This allows you to turn wordwrap on or off. The initial state is configurable, and it can be activated via command line. You should also make sure that the maximum line length for QD is always greater than the right margin, otherwise QD will handle wrapping lines as before (it puts a '↵' in front of the next line).

The wordwrap always looks at the current cursor position. If the cursor is left to the right margin, then nothing will happen. As soon as the cursor reaches the right margin, the current word and everything to the right of the cursor will be moved into the next line. If you move the cursor manually beyond the right margin and you start typing something, QD's word wrap becomes active! To summarise it in short: wordwrap tries to make sure that the cursor honours the right margin.

Tab Compression

QD can automatically convert spaces into tabulator characters (CHR$(9)) wherever possible on save and converts them back into spaces on load. It can be configured to do it on all files, or on specific files, depending on the file extension. When you pop up the F5 - Status menu and select the Tab options item, you will find yourself in a menu which looks very similar to the Extension Select menu which you also find in the File-Select window. The QD menu presents you the same eight extensions as the Extension Select does. QD reads the eight pre-defined extensions at the time when it is executed. To change the extensions, you have to change them in the Extension Select window before QD is startet. You can permanently change in in the Menu Extension by configuring it, if you want. But, back to the Tab Options menu which you find in the status menu. When you select "All files" here, all eight extensions become unavailable, 'cause they don't matter. When you deselect this item, you can select any of the eight single extensions. Every file which is loaded or saved which filename ends in an extension which is selected is automatically expanded or compressed, respectively. The default setting is that extension 1 (usually _ASM) is selected, the others are deselected.

The menu item at the bottom of this window makes sure, if selected, that only two or more spaces are converted into a tab character, if possible. Single spaces are not converted. This makes the resulting file much more readable from other programs like file viewers.

The compression does not slow down QD, it speeds it up effectively. In addition, files usually will become a lot shorter. Our assembly sources, for example, are between 30% and 40% shorter. This saves a lot of space on the floppy/harddisk and it speeds up assembly time considerable!

Before you active tab compression you should make sure that any program which will process the text file is able to treat tab characters as well as spaces as separators or "white spaces". This is true for some compilers and assemblers, but not for SuperBASIC! QMAC allows this feature from version 1.00 onwards.

Tab compression and decompression can be very helpful for other things. It can also be useful to re-format a text. If we assume that a source file is formatted with a tab distance of 9 and you want to change this to a tab distance of 8, it would be a lot of work to do it manually. QD can do it. Enable the tab compression, set the tab intervall in the status menu to 9 and save the file. Then, change the tab intervall to 8 and load it again. That's it, most of the work is already done!

Do not worry if you compress a file and the compiler/assembler does not understand tabs. Just load it again into QD, disable compression and save it. As long as you load the file with the same tab setting which was set when you saved it, your text file will not be modified at all as long as it is in the QD.

F2 - FILES

The F2-menu contains all commands which handle file load, save and print. You will find some red abbreviations after the menu items, e.g. ^L means [CTRL] [L]. To invoke commands or sub-menus which have those abbreviations, you do not have to go through the F2-menu, you just have to press the shown keystroke in the edit window.

LOAD TEXT FILE ..

Loads in a text file. Works as described above. You can invoke the load menu from the main edit window by pressing [CTRL] [L].

INSERT TEXT FILE ..

You are again asked for the file name. After you have entered it, the window will disappear and you get a cursor which shows an arrow pointing up and down. This tells you that you have entered a special mode- insert in this case. Move the cursor to the position at the text where the file should be inserted. If you press [SPACE], [ENTER] or one of the mouse keys, the file will be inserted at the current cursor position. This function will also unmark the block if it has already been marked. If you change your mind mid-operation, you can press [ESC] at any time; this will recall the normal cursor and you can continue typing as usual.

INSERT SCRAP

This function works in a similar way to the previous function, but it inserts the contents of the Scrap instead of a file.

FORGET CURRENT TEXT

Once again you will be asked if you really wish to abandon the current text. If you so wish, select YES and the current text vanishes. [ESC] or NO cancels the operation. Bear in mind that if you select YES, there is no way of recalling text if you have not already saved it!

SAVE TEXT FILE

Saves the current text to a file. When BACKUP in the STATUS-menu has been selected, the old file will be renamed by appending a _BAK to the filename. If there was an old _BAK file with the same name this will be deleted without any notice. If BACKUP is deselected, the old file will be overwritten. You will be asked for confirmation before the file is overwritten.
After saving, you can edit the text from the point it was at before it was saved. The filename will be put into the hotkey buffer, so that you can use it to load this file into another program by just pressing [ALT] [SPACE]. The same applies to WRITE BLOCK TO FILE and makes block transfer between concurrent QDs much easier. If your device driver of the device you save the text to is a Level 2 device (e.g. QL-Emulator), then the file version is also updated. Short selection keystroke from main menu is [CTRL] [V].

SAVE TEXT FILE AND QUIT

Saves the text, does not ask for overwrite confirmation and quits QD. Select this from main menu with [CTRL] [Q].

SAVE TEXT WITH NEW FILENAME ..

This function works in the same way as 'SAVE TEXT', except that first a filename is requested, or an existing filename is offered for editing. Main menu selection is done with [CTRL] [N].

PRINT ..

You can print the text or the current block by saving it with SAVE TEXT of course, but this will offer you a filename. PRINT TEXT offers you a printer port device name which you can configure with the CONFIG program supplied with QD. Here you can enter names such as SER1 or PAR_100k. If you select the device item with the right mouse button, the device select menu appears.
Next to it, you can define the number of print copies.
You can abort printing at any time while in progress. If you set the Form Feed option, a form feed is sent to the printer after printing. OK starts the printout. Print can be invoked with [CTRL] [P] from the main menu.
Printing to a file does not overwrite the file if it exists, it adds it to the end of the file instead.
Within print, you have two options of printing.
The "Direct (ASCII)" option works in the same way previous versions of QD worked:
You can define preamble and postamble strings (which you can pre-define with Config). The preamble string will be sent **before** the text is sent, the postamble-string is sent **after** the text has been sent. If "Form Feed" is selected, this will be sent as the very final instruction, after the postamble string. If the text is printed to a file (not a serial device like SER or PAR) then both the preamble and postamble strings are suppressed. The strings may be a repeated number of the following elements, all separated by commas: decimal values, or characters, which have to be introduced by a " or ' character each.

Example:

```
    27,64,27,"Q,80,27,'M
```

will send a reset to an EPSON-compatible printer, set the right margin to 80 (useful for a wide A3 size printer) and selects Elite (12cpi).
You can print via "Driver". For more details on printing via Driver and how to create drivers, see "Printer Drivers" in this manual. Drivers can only be used in SMSQ/E.

SAVE BLOCK ..

This command too should be self explanatory if you already know SAVE FILE. You are asked for a file name as usual to which the block should be written. If the file exists, you will be asked if you wish to overwrite it. That's all.

F3 - COMMANDS

The F3-menu contains most of the important control commands. You will find some red abbreviations after the menu items, e.g. ^G means [CTRL] [G]. To invoke commands or sub-menus which have those abbreviations, you do not have to go through the F3-menu, you just have to press the shown keystroke in the edit window.

GOTO ..

You can go to the start of the text if you select START or to the end with END. You can go to a specific line which you may enter. You can go to the current start of the block or to the position of the current marker. To select a line number in the GOTO menu, just type in the line number, the item is selected automatically. Select OK to go to the currently visible line number.
The menu items BASIC PROCEDURE and BASIC FUNCTION allow you to get a list of the Functions or Procedures if a SuperBASIC program has been loaded. All you have to do is select a name and you immediately get to the line where this proc/fn is defined. ASSEMBLER LABEL offers you the same facility for machine code sources. If you have configured a string for the user-defined list, then this item will also be available. It will give you a list of all lines containing the user-defined string, or better said the remaining bits of the line after the string occurence.
You may invoke the GOTO sub-menu by pressing the keys [CTRL] [G].

MARKER ..

A marker is a position somewhere in the text. You may select one of four markers by pressing [1], [2], [3] or [4]. The selected marker is the current marker. You may GOTO the current marker or you may set the marker. If you choose to set the marker, the cursor changes its shape to the marker number. You may now place the marker in the same way as you handle other special cursor (see INSERT FILE). After loading QD all markers default to line 1. [CTRL] [O] brings up marker menu from main menu.

SEARCH STRING ..

Searches for a string. If you have searched for strings before, these strings will be offered as the new search string. You can have up to four different search strings, but the one which has been recently edited or selected is the one QD searches for. You can enter and edit the string in the same way as file names. If a word is selected in the text, you can search for it by selecting TAKE WORD. You can select whether the search should go upwards or downwards. If you select MATCH, the case of the characters has to match, otherwise case is not important. If the settings are all correct, just press OK to start! If you wish to find the next occurance of the string, press [F9]. If you move the pointer to the "Continue Search" icon do not stop over a character in the text, as this would be the new position to search from. If the search fails there will be a short beep. You may select this menu by pressing [CTRL] [S].
Inside the search menu, you can press [CTRL] [R] or F3 to turn it into a replace menu.

REPLACE STRING ..

You will be offered the strings used previously for the replace, the same way as the SEARCH STRING is offered. If a word is selected in the text, you can search for it by selecting TAKE WORD. You can select whether replace should go upwards or downwards, input the number of times the replace should be done. ONE selects one, ALL all. When CONFIRMATION REQUEST is selected, a small window will be pulled up before every replace as near as possible to the string. You finally select where you wish to start the search from (e.g. FROM CURSOR)! You can press [ESC] to abort. Press [F9] to repeat the same REPLACE operation. Short selection is done by pressing [CTRL] [R].
Immediately after you replaced strings (one, more or all) the number of replacements is shown in the subwindow which usually displays the cursor row and column. After 2.5 seconds, the window switches back to the row/column display.
Inside the Replace menu you can press [CTRL] [S] or F2 to turn it into a search menu.

DELETE CONTROL CODES

Removes all control codes in the text file. If you load a text written with another editor which contains Carriage Return CHR$(13) for example, these will be removed. TABS CHR$(9) will be expanded with spaces as defined in the STATUS menu. All other non-printable characters will be deleted. There is no short selection keystroke as this option is very unlikely to be used.

CHARACTER TABLE ..
allows you to change the character under the text cursor. You get a menu with the complete character set in a table. On the top you see the current selection, which is initially the character under the text cursor. Below the table, you see the current character (which is the one under the pointer). If you press [SPACE] or the right mouse button, this character is moved to the top and assumed to be the current. Select OK now, and the character in the text is changed to the current selection. If you press [ENTER] or the right mouse button while you are over a character in the table, then this character is returned. Short selection is done with [CTRL] [T].
The character select menu remembers the character which is selected. The next time you pop up character select, the current character is displayed as a selection - press [ESC] to leave it as it is. The pointer appears over the character which you selected last time you used character-select, so you just have to press [ENTER] to insert it.

LINE NUMBERS ..
pops up another sub-menu:
START is the number of the first line when line numbers are added. STEP is the line number increment. ADD adds line numbers to the current text, DELETE removes line numbers. RENUMBER is ADD and DELETE, it does not renumber BASIC references.

ORDER LINES ..
orders the lines. You can specify the sort criteria (column range which is used for determining the order). You can order ascending or descending. Please note that ordering a large text can take quite a while!

TYPING CHECK
makes QD look for the spell-checking extension QTYP_SPELL the first time it is invoked. It tries to open a SPELL channel. If you already used QTYP your machine will probably hold a dictionary, otherwise SPELL will try to load in a dictionary. If it does not find a dictionary, QD will tell it to you. If a dictionary is found, all words are checked starting from the current cursor position. If a word is not known, a window appears showing the offending word with the request whether you like to add it to the word list (learn it) or not. Press [ESC] here to abort type-checking. As soon as you start typing check, an 'Off'-Option becomes visible in the F3-menu. This item allows to close the SPELL channel opened by QD. If you wish to change or remove the main dictionary, you have to turn Typing Check off in every QD which is using SPELL.

QUIT QD
This is the best way to leave QD. You have to confirm as leaving QD also removes the text. [CTRL] [X] quits QD from main menu.

F4 - BLOCK

First we should define 'block'. As you see, the text is normally displayed in white on a black background. A marked block is a part of the text which normally has a start and an end. This part, called the block, is displayed green on black. You can edit the text within the block, of course. This will be added to the block the normal way.

There are some ways of defining a block. The easiest is doing it with the mouse. Just move the cursor to the point you wish to mark the start of the block and press the left mouse button. The block start is set, the following text gets green and the cursor changes to the 'mark end of block' cursor. NOTE: this time the left mouse button does not the same SPACE does! You can now mark the end of the block, and the cursor turns back into its "normal" state.

You can also use [F7] to mark the start of the block (see F7). If no end has been marked before, the end of the text will be taken to be the end of the block. If you wish to delete the block, nothing will happen until you have marked both the start and the end of the block. This will protect the text and the block.

You can also use [F8] to mark the end of the block (see F8).

FORGET
Forgets the start and end of the block, there is no longer a block marked. Note: This will not delete the block.

DELETE
Deletes the whole currently marked block. You have to confirm this command. The block has to be marked with both start and end, otherwise you have to call DELETE twice. The short-selection keystroke is [CTRL] [K].

GOTO START
This command simply does what you think it should do: it moves the cursor to the start of the block. If there is no block marked, nothing happens.

COPY
Puts a copy of the marked block to another position and marks the block at that position. If you select COPY you will again get a special cursor, the same as you got when you selected INSERT FILE (the function is nearly the same). Move the cursor to the position to copy the block to (it may be within the same line), press [SPACE] or [ENTER] to copy or [ESC] to cancel. You can activate the copy cursor by pressing [CTRL] [Y] in the main menu.

MOVE

Behaves the same way as copy does, but deletes the block at its old position. Main menu selection is possible with [CTRL] [U].

OVERWRITE SCRAP

This function puts the current block into the Scrap and overwrites its previous contents. You will find the opposite command to read the contents of the Scrap in F2 - INSERT SCRAP. You can imagine the Scrap as being a RAM Disk which may contain only one file. The Scrap should be used to transfer any kind of data (in this case Text) between programs.

ADD TO SCRAP

This function adds the current block at the end of the current Scrap contents. This allows you to collect different parts of a text, add it directly in the scrap and load it as one part in the same or another QD.

UPPERCASE

Just does what it says: it changes all lower-case letters within the block into upper-case.

LOWERCASE

The opposite of UPPERCASE.

BLOCK SHIFT

Can be used to shift the block. First, you should define how many characters should be shifted. Next, select the fill-character or string if you plan to shift right. The new space will then be filled with the fill-character (a quick way to comment assembler text out, for example). If you shift right and insert a single character, the single character is repeated and inserted in the freshly created space. If you enter more than one character as the fill string (e.g. "REMark") then the space will be filled with the string, padded with trailing spaces (provided it fits) or part of the string (if the space is smaller). Then, select "Left" or "Right" to shift the block. The new block will be set from the leftmost character of the first block-line to the rightmost character of th last block-line. The block has to have a proper start and end, otherwise shift will not work.
[↑] or [↓] brings you from the number of characters to shift to the fill string and vice versa. [ENTER] finishes the input, as usual.

F5 - STATUS

Here you can have a look at the current configuration of QD, you can edit and alter it if you wish.

TAB INTERVALL

In this menu you can also set the tab stops. The cursor will skip to a multiple of this value if you press the [TAB] key. Example: Enter 15, go back into the text and press [TAB]. The cursor will skip to the next multiple of fifteen. If your cursor is left, the next [TAB] skips to 15, from position 41 the cursor will go to position 45 and so on.

LEFT MARGIN

You may also set the left margin. This is the position at which the cursor should come up if you press [ENTER]. This is very helpful when you enter assembly code. Set your margin to 8 or 10. Set the same value for the tab. Enter your program text as usual and only if you need a label press [SHIFT] [TAB] to go back to the beginning of the line. A special case of the left margin is provided for fans of structured programming: Enter a blank (an empty string, not 0, of course) and you get soft margins, i.e. the cursor will be placed at the same position as the start of the line above. You can move the cursor left or right, the next line will have the new margin.

RIGHT MARGIN

The right margin does not do a lot in QD, except for generating a warn beep if you press a key while you are right from the right margin. If you turn the Wordwrap icon on (see above) it will do a half-automatic word-wrap, but this is it more or less.

FILE EXTENSION

You will probably work with many files which have the same ending, for example _ASM, _PAS or _BAS. You can enter here the ending of your choice. Now you can omit this ending when entering a file name. If you also have DATA_USE, the program's search for the file goes like this: First the file with the file name you have entered is searched for. Let us take DEMO for the file name. If you have entered a complete name, say flp2_DEMO_BAS, this will be found and loaded. If you have only entered DEMO it cannot be found. So the DATA_USE device name will be placed before it. If DATA_USE is FLP1_, QD will look for a file called FLP1_DEMO. If this fails, DATA_USE will be removed and the ending will be appended, e.g. _ASM. Another search for DEMO_ASM takes place. This would also normally fail. DATA_USE will be put before the file name and finally FLP1_DEMO_ASM will be looked for. If, in the end, this does not exist, an error will be reported. Tricky, isn't it? But this saves you lots of keystrokes if you use it properly. If you DO on this item the current extension is deleted and you immediately return from the menu. If you

DO on this item and there is no extension, you will be presented the Extension Select menu.

DIRECTORY
You may set the working directory of QD. The contents here depend on what you configured. DOing on it will clear the contents - this means that the DATA default will be used next time the file-select window appears. If you DO on an empty item, the directory select window appears.

HELP FILE
shows the name of the file which would be displayed if you hit [F1]. Of course, the help file does not necessarily has to be a help file, it can be anything worth to have a quick look at, e.g. key files etc.

FONT LOADED ..
Loads in a QL font and installs it as the font which will be used for displaying text in the text area and the main window. You can load special fonts which also display codes 0 to 31. The supplied QD_fnt may be used if you wish. Note that all font file names have to end with _fnt. Fonts must be in the QL standard format; you cannot load fonts such as those supplied with QWriter II or the hires fonts of PAGE DESIGNER.

CONFIRMATION REQUEST
If this item is deselected, you will not get any YES/NO requests. Exceptions are errors, they will always be reported. An exception to the exception is the 'file exists' error. There will be no request and the file will be overwritten without any notice. The REPLACE request will also come up if selected.

BACKUP FILE ON SAVE
Have a look at [F2] SAVE FILE.

KEY-CLICK
Allows you to enable or disable a key-click which sounds every time a key is pressed within the editing window (except for selecting one of the functions of the top area).

TOOLBAR
Allows you to turn QD's toolbar on and off. This will give you two additional lines.

USAGE..
It is possible to highlight lines in QD. The highlight depends on the usage. You can configure QD to pre-define a usage or use the parameter string option "\U" to set the usage (see above, parameter string).
At the moment, the following different usages are possible:
General: QD behaves as before.
Assembler: All comment lines (lines starting with * or ;) will be highlighted.
BASIC: All lines containing DEF PROCs or DEF FNs will be highlighted.
Messages: All lines with ">" in column 1, 2 or 3 will be highlighted.
User-Defined: Will highlight all lines containing the user-defined string. This is a very comfortable (and different) way of finding string occurances in the text. Give it a try and see how useful it is.
Refresh highlights updates the highlights. There is no automatic update, as this would consume far too much processor power on slow systems. Please note that the highlight is not perfect. It is quite a late add-on to QD and was never planned when QD was initially designed. Therefore, panning may result in strange colour effects.

F6 - WORD

On an ordinary QL you do not have the [F6] key, so you have to press [SHIFT] [F1]! But, first of all: what is a word? It must not necessarily be what you understand by saying 'word'. A word may be any group of characters, separated by a group of other characters. The separating characters may be pre-set with CONFIG, or you may set it in F6-Word delimiters. SPACE is always a delimiter. All delimiters are used if you jump wordwise (or delete). It is sometimes quite useful to be able to define whether, for example, a minus delimits two words or not.

Put the cursor over a character within a word and then press [F6] **-** [S] to select the word. You may also use the single-keystroke selection [CTRL] [W]. The word is marked to be the current block.

INTO HOTKEY BUFFER stuffs the current word into the HOTKEY Buffer. This also works on a block covering up to a complete line. As soon as more than one line is marked, nothing is stuffed.

You will see many conversion items to convert a word (preferably a number!!!) into a number of different base. If you select something, you will see the REPLACE window filled in with the search string of the selected word and the replace string of the converted value. Conversion is currently only 16 bit long!!!

This menu also shows the character count for the current word.

F9 - AGAIN

Repeats the last FIND or REPLACE command. The QL needs [SHIFT] [F4].

F10 - Thing

We already mentioned it in the manual above: You may specify a Thing name which appears behind the F10-option in the main menu. You may either configure it or pass the name with the \T key in the parameter string. Further information about things can be found in one of the further chapters.

The CONTEXT Menu

The CONTEXT menu can be invoked with the right mouse button. Depending on whether the cursor is positioned over text, some or all of the items are available in the menu:

SELECT WORD

Selects the word under the cursor. Same as F6 - Select Word .

... AND GET HELP

Selects the word and calls the help system.

... AND SEARCH

Selects the word, opens the Search menu and inserts the selected word as the search string.

LINE INTO HOTKEY BUFFER

Stuffs the current line into the HOTKEY Buffer. It can be brought into other programs by pressing [ALT] [SPACE] when you are in the destination program's cursor entry).

BLOCK START

Same as F7.

BLOCK END

Same as F8.

FORGET BLOCK

Same as F4 - forget block.

The Menu Extension

Many sub-menus will contain windows which look similar, e.g. for selecting files when you load any kind of files. These windows are not defined within QD, they are in the Menu Extension. This means, replacing the Menu Extension can make selecting files much more comfortable (just an example, secting files is nearly perfect!). This means that the following chapter deals with the Menu Extension and every reference to configuration does not mean, that QD has to be configured but the Menu Extension (MENU_rext).

Just for a quick reminder: Hit means, point to an item and press [SPACE] or the left mouse button, Do means, press [ENTER] or the right button while you are over an item.

The file-select window

The file-select window is always shown when the user is required to enter or select a filename. Here you can enter the filename either directly or edit a suggested one by selecting the menu option directly beneath the request. Beneath this are two menu options with which you can recall the contents of the HOTKEY buffer and all previous contents. Just select the menu option and the contents will be written to the "suggested" area. Confirm with OK and the input will be accepted by the system. You can also edit the name, of course. Select it with [F3] or point to it and press [SPACE]. When the cursor flashes, you can edit the string. [ESC] brings you back into "pointer mode".

The rest of the window concerns the current drive. You will see two sub-windows, the right hand (larger) one will only show the filenames, the smaller one on the left only the subdirectories. In these windows all the files are sorted alphabetically (unless you configured the sort off). The files will be taken from the current drive and must all have the correct extension, or ending (if any). The extensions for sub-directories are ignored, because subdirectories don't really have extensions. Now you can edit the drive and/or the ending. Depending on how you configured the Menu Extension, you will either get the full (complete) file- and directory-name, or just anything which comes behind the current directory or device name (e.g. if the current directory is WIN1_TEXT_ and it contains the files WIN1_TEXT_FRED and WIN1_TEXT_PAUL, only FRED and PAUL will be shown).

You can get quickly into the file-list subwindow by pressing [TAB], or into the subdirectory-window by pressing [SHIFT] [TAB].

If you press [F2] or [ENTER] at the directory menu option, a further window is overlaid, from which you can select pre-defined devices and sub-directories (see Directory-Select below). If you just select a directory, the list of files will be updated. You can also "update" it by selecting the lightning symbol or by WAKEing up the job to which the window belongs, should this be hidden.

If you press [UNDO] or [CTRL] [ALT] [↑] then the "Ext:" menu option will be cleared, so that all files will be shown. If you press [ENTER] at the "Ext:" menu option, then a window overlay will show some suggested extensions (see extension select below). If you press [F4] you will get directly into the extension select menu, regardless of the current extension setting.

If you press CTRL 0, CTRL 1 or CTRL 2 (does not work on any keyboard layout, sorry!) then you will see TYPE 0, TYPE 1 or TYPE 2 instead of an extension. Only those files are listed which match the filetype: TYPE 0 is used for most files including text files, TYPE 1 shows executable programs only, TYPE 2 all relocatable files (which usually end in _REL).

If the extension starts with an underscore, then this underscore is automatically internally converted into a "." for DOS disks (e.g. the extension is _TXT, but if you look at a DOS-Disk, it will list all files ending with .TXT).

Above the current directory is a list of available devices, e.g. WIN or FLP. There are also drive numbers from 1 to 8 listed. To select FLP1_, press [F] and [1], and the directory window displays FLP1_.

Above the file list you will see an arrow "<-". By selecting this option you can retrace a step back along the subdirectory tree without having to edit anything. So if you're in directory FLP1_TEXT_PAUL and select this once, then you get to FLP1_TEXT_. The next time you select it, you get FLP1_. If you select a subdirectory in the lefthand subwindow, then you'll "get in", i.e. the name will be taken over for the directory and the file and subdirectory list read in again.

But to get back to the list of files: You can select any file you like. [SPACE] accepts the name as "suggested" and enables the cursor, so that you can edit it. [ENTER] takes the name and carries out an OK automatically. If the window is too small to show all the suitable files, the normal scroll arrow will appear in order to scroll the next batch of names up the screen. You can also select files or directories by pressing the character which is in front of the name.

At the right you will see a scrolling bar. Move to this area, press [SPACE] and the area will be shown relative to the total area. Press [ENTER] and the window will split, enabling you to control the two parts indepently from each other. Move to the split and press [ENTER] to join the window together again. If you are already in the filelist-subwindow and you press [TAB], the cursor jumps over the bar. Another [TAB] brings you back to the centre of the filelist. For the directory-subwindow, [SHIFT] [TAB] does the same.

The file-select has got two other facilities: you can get a list of the whole directory tree, i.e. all sub-directories from the current level downwards are "expanded" into all filex they contain. This gives you a better overview. But beware, the tree option may take a long time, especially on higher levels on heavily filled harddisks. Therefore, tree is de-selected automatically when you "move up" in the tree.

You can also have a quick look at the file before you select it for load. Select the View item first, then hit the file you want to view. You can view it now until you press [ESC]. If you think this is the right file, select OK 'cause the filename has moved to the suggestion. You can also press [ENTER] while you are over the filename, as [ENTER] ignores the state of the view item.

You can construct a new filename very easy: choose the required device and subdirectory in the left part of the window. Then, make sure the filename extension is right. If you press [F5] now, the current directory and the current extension will be used as the "suggestion" and the cursor is placed exactly between directory and extension. All you have to do now is: type in the missing part of your filename. If you would like to save a file called "win1_source_qd_fred_asm" and you are already in the directory "win1_source_qd_" and the extension is "_asm", then just press [F5] and type in "fred".

Every filename selected with [ENTER] or [OK] will also be stuffed into the HOTKEY Buffer, so that it can be immediately selected from somewhere else, or the next time you pop up a file-select window.

The directory-select window

This window allows the selection of different drives resp. subdirectories.

The user can pre-set the eigth most used subdirectories; they can be selected in the left sub-window. Use the MENUCONFIG program to pre-set the subdirectories on the MENU_rext file. If one of the preset directories is hit, it moves to the suggestion at the top of the window. If it is DOne, this selection is returned to the calling program. If the selected subdirectory does not exist, the device is searched for matching names in higher levels. If, for example, you select FLP1_BACKUP_TXT_ and there is no such directory on FLP1_, then only FLP1_ is shown. If the directory BACKUP exists, you will see FLP1_BACKUP_.

The currently installed drive systems (MDV, RAM, FLP or WIN, for example) are selectable immediately, together with the eight different possible drive numbers. If selected, the suggestion changes and the list of subdirectories is read in again.

The right sub-window displays the subdirectories which exist in the current directory level. This window behaves exactly in the same way as the subdirectory-window in file-select does.

[F2] allows you to edit the current suggestion.

"DATA Default" sets the suggestion to the current setting of the system's data default directory, which is usually set using the DATA_USE command, or QPAC II's SYSDEF, for example. The same applies to "PROG Default".

To configure the subdirectories please use the program MENUCONFIG and configure the file MENU_rext.

If the Menu Extension itself is not built into ROM or EPROM, but loaded into RAM, you will see an "Edit" item being available.If you select this item you can edit all of the eight pre-defined directories and save a new configured version of the Menu Extension directly afterwards. Press ESC here if you do not want to save it, or give the filename. If you modify the suggested filename for the Menu Extension, this modification is also saved together with the new preset directories.

The extension-select window

This window serves to select the most usual file extension. Move to a menu option and press [SPACE] or [ENTER] to select. You can also use the number keys [1] to [8] to select the menu options. All ten extensions may be set by the user using MENUCONFIG, or, if the Menu Extension is in RAM, by selecting the Edit item - it behaves exactly like the Edit facility in directory-select (see above). 'No' returns no extension, i.e. an empty string.

The HyperHELP Helpsystem

The help system only works if QD is installed resident. QD contains help facilities which are very flexible and can be used for any type of help. It is possible to use the HyperHELP system for BASIC directly, which means you get immediately help on SuperBASIC procedures and functions (i.e. explanation, syntax description and examples) - this already comes with QD! Help in assembler, for example, could be that if you require help on the definition of library routines or sub-routines used in other modules, you get the source definition of the sub-routines. Helpful, as you cannot remember all the time which register are used for parameter passing etc. Although this type of help is user-dependent, a program which "collects" entries of machine code libraries is enclosed with QD. But help can be completely different, it depends on you. Theoretically, every word used in any QD text can be asked for help and it depends on you whether you want to support help or not.

Before the help system was developed, we thought about how to design it. The most difficult thing was the decision how the help information should be stored. The full help could be contained in one large file, giving the advantage that only one file needs to be copied and it could save storage space But, this would result in two major disadvantages: special tools would be required to add and/or remove help on other software packages and it would be very difficult for the user to add own examples and comments. In addition, scanning through a large file for help subjects could slow down things if several kilobytes have to be searched. Therefore, we decided that every explanation for a single item (or a small number of items) to be put into separate help files, which are simple ASCII text files. Of course, a directory or "INDEX" file is required to tell the help system where to find the right help. This means, that the same file can be used in different help systems and that one or more help subjects could be indexed to the same help file. In addition, every help file and the index file can be modified with QD. New help files can be added easily and adding new help subjects is easy as well.

On to the index-file: it is called HELP_INDEX, and it must reside in the help directory which is either the pre-configured one or the directory passed in the parameter string. Let us have a look at the HELP_INDEX which is supplied for the BASIC help. Load it into QD and look at it.

Every line contains two words: the first is the search subject, the second is the filename which contains help on this subject. They are separated by one or more spaces. The filename can be either the full filename or just the part of the name which comes behind the help directory (i.e. the directory which contains the INDEX file). This means, it can be either FLP1_BASIC_HELP_PRINT for a full filename or just PRINT if the file is on FLP1_BASIC_HELP_, for example.

It is advisable to sort the search subjects in alphanumerical order, i.e.

```
a
aa
aab
```

```
ab
ac
```

A concrete example would be:

REP (which stands for REPeat) should come before REPORT which should come before REPORT_ERROR.

Having an alphabetical order gives two advantages: it is easier to find a name if you search through the index file manually, and you make sure that you do not use the same subject twice. It also makes sure that you do not get help on a wrong subject. When HyperHELP searches for a subject, then the index file is scanned line by line and the search subject is compared with the beginning of the first word in every line. If it searches for "REP", for example than it thinks it is found as soon as it finds a wort "REP" or starting with "REP", which means "REPORT" would be treated as found when it comes first.

You have to tell QD where to look for the help index file. You can pre-configure it (using CONFIG or MENUCONFIG), which means, you get the help on every copy of QD which is executed. You can change it while QD is running in the status menu. If you want to start only one QD with specific help, then you can pass it in the parameter string with the \H parameter. You see, there are many options. Choose the one which you think is most suitable for you.

If you edit a text or program in QD and you require help, simply move over the word and press [F1] or [HELP]. Another QD will pop up which contains the file which is defined as the help index file to be the help on the selected word. If the word is not found in the help index file, you will see a standard help text, found in "HELP_TXT". The new QD can be used as any standard QD, with the exception that the text cannot be modified in the beginning. You can scroll through it and move the cursor, but any input is ignored and every deletion keystroke moves the cursor only. As it may be desirable to change the help text, add examples etc. you can "unlock" the entry by clicking on the three --- next to CF6. The item will change to the standard OVER or INS mode, and you can edit the text freely. Of course, help on subjects within help is available and so on. Every Help-QD is not called "QD filename" anymore but "HELP filename" to allow different hotkeys for picking. Every Help-QD can be left with the [ESC] keystroke quickly.

Help for Assembler

Especially in assembler it is helpful if you get the definition of a subroutine on a single keystroke - nothing is better than the well documented assembler-source-listing. The SuperBASIC program MAKE_INDEX_bas helps you to create help index files. Load it into SuperBASIC and edit the first few lines. Line 120 should contain the directory where the HELP_INDEX is to be created. Lines 160 onwards contain the directories to be searched. Delete all the defaults and insert your own directories. The list should be terminated with an empty string. If you RUN the program, all the "_ASM" files found in the specified directories are searched for external definitions (XDEF) which are put into the HELP_INDEX file. It's simple, try it! Remember: qliberated works faster!

HyperHELP for BASIC

HyperHELP is a HelpSystem for SuperBASIC. Maybe you are one of the users who cannot remember all the complicated syntax and different parameters required by many SuperBASIC procedures and functions. No wonder, if you add SuperBASIC extensions which are todays standard then you end up with 500 or 600 commands or functions. Another problem is: where do you find the relevant explanation which belongs to a specific procedure or function? You will probably have to search through many manuals! That's over now, HyperHELP helps!

HyperHELP can be installed in different ways, it depends on your hardware and also on how often you use SuperBASIC and require help. Anyway, you should now make a working copy of the HyperHELP master disk before you continue to read! Ready? Fine, let's continue. Now choose one of the following options how to install HyperHELP in your system:

1. You own a harddisk (best case, of course): create a subdirectory in which you copy all the help files. HyperHELP has been pre-set to FLP1_BASIC_HELP_. If you create a directory on your harddisk WIN1_BASIC_HELP_, then all you have to do is copy all the files from floppy to harddisk.

2. You have one or more floppy disk drives, one of which can be used for a HyperHELP disk during the session. Great, simply leave the HyperHELP disk in this disk drive!

3. You do not have a spare disk drive, but you have enough RAM in your machine (i.e. GoldCard or QL-Emulator for the ATARI). At the start of your session, copy all the help files from the floppy disk to an rarely used disk, for example RAM5_ or RAM6_.

4. You do not have a harddisk, a number of floppy disk drives or a lot of RAM. Then you have to insert the HyperHELP disk in a floppy disk drive at any time you want to get help.

If you still think you loose too much space (either harddisk space or RAM space) then simply view the different helpfiles and delete those (not on the master disk) where you think you do not need the help because you know the command very well (things like LET, for example). Experience has shown that it is not a good idea to delete those which you think you do not need, because if you need it then you have to search, and that's where HyperHELP should be able to help you instead of searching through manuals.

Regardless on your decision, you have to tell HyperHELP where to search for the index-file which contains the help index. You can leave it as it is (it was already mentioned, default is FLP1_BASIC_HELP_) or configure it, or do it with a parameter string:

```
EXEP flp1_QD;'\h flp1_basic_help_ \e _bas'
```

starts QD and sets the help directory to FLP1_BASIC_HELP_, and loads files which in in _BAS only.

Adding new Help-files

How do you expand the help files? If you want to create own help texts, then write them in an editor or Quill, text87 etc. and save them into files. Then load the HELP_INDEX file and add a line containing the help subject, a space and the filename of the new helpfile. That's it. If you want that a number of subjects refer to the same help file, then create a line for every subject.

If you own other Jochen Merz Products, then you will see that all these programs which contain SuperBASIC extensions will be updated so that help files on the commands will be supplied (it will take some time, but it will come). Copy all the help files into your help directory on the harddisk or onto your help floppy disk, and then load the HELP_INDEX file into an editor. Merge the file HELP_INDEX_ADD at the end of the HELP_INDEX file and save it. That's it. If you want, you can sort the new help subjects into alphabetical order.

General Help features

If the help system is activated you can activate help by pressing the right mouse button. If at this time a block is being marked, then it is forgotten first, and you have to press the right mouse button once again to activate help.

If you select help about a word, then this word is used in the "help QD" to be used as a search string. Therefore, the cursor will be positioned over the first occurance of the word. And, even better, you can carry on searching for the help word by pressing [F9]. This feature is used by FiFi too, but you need version 2.10 or higher. If you are searching for a word and you have configured FiFi so that it uses QD, then QD automatically searches for the word FiFi was looking for. This sounds more complicated than it actually is, just try it!

Printer Driver

The current version of QD allows you to print via printer driver. It requires QD to run under SMSQ/E. "Another printer driver", you may think!? Yes, but we tried to make it as flexible and as easy at the same time. And, you don't have to use it if you don't like it.

The printer driver can be a simple SBASIC program (called "filter"), which just translates a few characters or reacts on simple codes like "Bold" or "Italics". It can be very complex, if you like - you just do it yourself. It can also be compiled BASIC, and, of course, a filter written in C or assembler.

The concept of a filter is very easy to understand: When QD normally prints, it prints directly to an output channel, e.g. PAR or SER or a file. When a filter is activated, the output is sent to the filter. It arrives in channel #0 of the filter, and it is up to the filter what to do with it. It can just forget about it, modify it or send something completely different. The filter prints to #1, from where it goes to the output device (as before, PAR, SER, or file).

You will find sample filters on the QD disk which give you an idea what a filter can look like. If you're not familiar with assembler, forget about the assembler filter. Have a look at the BASIC filter instead. It is designed for EPSON compatible printers. If you don't have an EPSON compatible printer, edit the control codes.

QD sends codes which are easily identifiable. If the filter does not "know" the code, it should just forget about it. This makes sure, that old filters can cope with more recent versions of the program, and that old programs can use new filters.

The codes start with CHR$(1), followed by a single-character key, optional parameters, and end with CHR$(2). The easiest filter you can think of, therefore, ignores everything between CHR$(1) and CHR$(2).

We have defined the following keys with the following parameters:

+	Preamble. No parameter. Supported by QD.
-	Postamble. No parameter. Supported by QD.
f	Filename of current file: parameter is complete filename Supported by QD.

P	select Pica (10cpi). Supported by QD.
E	select Elite (12cpi). Supported by QD.
C	select Condensed (20cpi). Supported by QD.
D	set Draft (or economy printing): 1 is on, 0 is off. Supported by QD.
H	set Highlight: 1 is on, 0 is off.
I	set Italics: 1 is on, 0 is off
B	set Bold: 1 is on, 0 is off
U	set Underline: 1 is on, 0 is off

For cell-based applications (like QSpread, but not QD), every cell is preceeded:

T	Text-contents
N	Numerical contents

The QD Things

As explained earlier in this manual, a Thing may be specified to be used in QD. The name of the thing appears in the F10 menu item. The Thing name may be pre-configured or passed to QD with the \T parameter.

QD can handle two different kind of things now, which do not look different from the outside but work in a different way. The first, old version (V2) only searches for a thing when [F10] is selected, does the action defined in the Thing and frees it immediately afterwards. Naturally, you can do nice things with it but not everything, it was not flexible enough. But, you'll find a V2 example thing on the QD disk, explanation follows down on this page.

The new definition, V5 Things, is much more flexible and works in a different way. When QD is started, it immediately searches for the Thing. This means, that it has to exist at this time and can do a lot more. It can pars or check lines when they are entered, call back actions in QD and much more, and of course, can be called with [F10].

If you are interesting in programming own QD things, you can get a definition of the structures and routines in the JMS-Mailboxes 0203-502013 and 502014. Of course, a printed documentation is available - write to Jochen Merz Software and send two international reply coupons for the postage.

You will find various QD things in the Mailboxes mentioned above as well.

SMSQ/E specific features

Some of the features built into QD will only work if you run it under SMSQ/E. This is not because we want to make you buy SMSQ/E, but it is just not possible with QDOS.

As SMSQ/E allows you to read and write from and to DOS and TOS floppy disks and harddisk, QD will auto-detect the file-format. QDOS only uses <LF> to terminate a line, whereas other operating systems need a <CR> character as well. QD detects automatically if the disk is an MS-DOS type disk and adds a <CR> automatically to every line on save or removes it on load.

Printing via printer driver is available only when QD is running under SMSQ/E, because some features which are not available in QDOS have to be used.

Configuration of QD

You can use the program MenuCONFIG to configure QD so that it will start next time with the defaults required. Start CONFIG or MenuConfig with:

```
EXEC flp1_MenuConfig
```

Note the Pointer Environment, the Window Manager and the Menu Extension have to be installed. Type in the name of the file which contains QD or select it from the menu (e.g. a hotkey or BOOT_REXT file) or QD itself, e.g.

```
flp1_QD
```

You can follow the program step by step, you can press [ESC] at any time. Then you can enter the file to which the modified version of QD should be saved. Please do not overwrite QD on your master disc/cartridge.

Note that if you configured QD in a hotkey or BOOT_REXT file, the configured QD will become active only after a reset.

You may define nearly every parameter which can be changed in QD. The configuration possibilities are grouped into four separate categories:

General/Windows

First you choose whether QD should appear full-sized or as a button after it is started.

Next you define whether the filename in the button (if QD is turned into a button) has got the full length or only the short name.

Then you can define how large the main window should be, i.e. how many lines the edit window should be high.

The maximum line length is always a multiple of 80. The longer the lines are, the more workspace uses QD. Normally, 160 should be suitable.

Next comes the origin of QD. It can appear at the current pointer position when started, or at a given origin.

QD always tries to put the button in the button frame, but you can give here the origin of the button if no button frame exists.

You may also set the colours of the QD windows. You can choose between four combinations, called colourways: green on white (the default setting), red on black, red on white and green on black. You can also tell QD to take the colours from the colour settings of the MENU_rext file.

Also, the text in the edit window can be defined to be white on black or black on white.

Next, you can configure whether QD should start with a visible toolbar or not.

Things/Files

First you define whether QD should install the FileInfo-Thing is loaded as a resident extension. This is required in order to be able to execute a text file from within QPAC 2's Files menu and invoke QD automaticall with this file.
The FileInfo Thing which is built into QD is extended, so that it can replace SMSQ's inbuilt FileInfo Thing. This means that you can stil execute _BAS and _SAV files from QPAC2's Files menu directly, or get into QD if you try to "execute" other text files.

You may pre-define a THING-name for the F10-option. Currently, there is no need for you to do this. You will soon see that there are many possibilities with the implementation of F10 - just read in this manual.

You may disable the HELP item by setting the help-file-name to a null string, i.e. no name. Most people do not need help and you occasionally hit the [HELP] or [F1] key. When HELP is disabled, [F1] has no function.

You may also give the name of a font file to be loaded automatically when QD is started. No string does not load anything.

You may set the working directory of QD to one of three options. The first one - 'Current DATA default' makes QD behave like before, it always uses the current DATA default (which means: no local directory). You can set it to 'Copy of DATA default', which is much more clever: QD always uses the DATA default directory which is set when QD is started. You may then set the DATA default to some other directory, but QD still uses the one which has been assigned to it. Furthermore, if you go to a different directory while loading a file, this change is also stored in QD and the working directory is updated. You pre-set your preferred directory for QD, which also gives you the option of changing it while QD is running.

Then comes the default extension, followed by the default print output device, and the default setting for the Form Feed button in the print menu. If nothing is specified for the default print output device, then the setting of DEST_USE when QD is started is used instead.

You can specifiy a printer driver and whether it should be initially selected or not.

Editor

First you may define the default tab distance, the left margin and the right margin.

Now you can select if the cursor is flashing or non-flashing.

Next, you can define if confirmation request should be on or off, and the same for backup on save, whether the scorll bar should be updated or not (saves time) and for the keyclick in the editing window.

The defaults for the line-numbers menu may be set now.

The next configurable item here is the 'List of Word Delimiters'. You will see what it does when you look at the new F6-Word menu.

The lists which are create by GOTO assembler label, GOTO BASIC function or procedure are sorted alphabetically now. This seems to be more practical, but this feature can be turned off by configuration.

You can also specify a user-defined string to create a GOTO List. The text is searched for the specified string when you select the command in the F3 - Goto menu (see F3 - Goto).

If a line is selected with [CTRL] [Z], it can be stuffed into the HOTKEY Buffer (and brought into other programs by pressing [ALT] [SPACE] when you are in the destination program's cursor entry).

The initial fill-string and number of shifts for Shift Block Right can be given.

You can set the initial state of the Wordwrap item.

SHIFT SPACE is the default keystroke to toggle between insert and overwrite mode. You can use CTRL CAPSLOCK instead.

Finally, you can define QD's usage for highlighting lines and a user-defined usage string.

Tab Options

Here you can define the options for the automatic tab compression. If you select All Files here, then the next eight possibilities are unavailable. If it is not selected, then you can select for all of the eight default extensions whether files with filenames ending with this extension should be automatically compressed and decompressed on load and save.

You can also pre-define the state of the "Keep single spaces" item in the Tab Options sub-menu. If selected, there must be more than one space between two characters to get it compressed into a tab character.

Bitte lesen Sie sich den Teil dieser Anleitung durch die die Tab-Kompression ausführlich erläutert!

Summary of Short Selection Keystrokes

In some pulldown-menus you will find red characters (e.g. ^L) after the menu entry. This means, that you can select this item by pressing [CTRL] [T] in the main menu without going through the F2/F3/F4/F6 sub-menu. This is called a short selection keystroke. Here is a complete list of all short selection keystrokes available in QD:

CTRL A	press it the first time to mark the start of a block, a second time to mark the end (same as left mouse button).
CTRL B	cancel block marking or forget the current block (found in F4) - (same as right mouse button).
CTRL C	switch to next job (QDOS function).
CTRL D	gives Help (same as [F1]).
CTRL E	Window Move (same as [CTRL] [F4]).
CTRL F	Window Resize (same as [CTRL] [F3]).
CTRL G	brings up Goto pulldown menu (found in F3).
CTRL I	Tabulate (same as pressing the [TAB] key).
CTRL J	Linefeed (same as pressing [RETURN] or [ENTER]).
CTRL K	deletes the current block.
CTRL L	brings up Load text file (found in F2).
CTRL M	inserts a Carriage Return characert (same as [SHIFT] [ENTER]).
CTRL N	brings up Save with new Filename (found in F2).
CTRL O	brings up Marker pulldown menu (found in F3).
CTRL P	brings up Print menu (found in F2).
CTRL Q	saves current text file and quits QD (found in F2).
CTRL R	brings up Replace menu (found in F3).
CTRL S	brings up Search menu (found in F3).
CTRL T	brings up Character Table menu (found in F3).
CTRL U	activates the Move Block cursor (found in F4).
CTRL V	saves the current text file (found in F2).
CTRL W	marks the word under the cursor as current word (found in F6).
CTRL X	quits QD (found in F3).
CTRL Y	activates the Copy Block cursor (found in F4).
CTRL Z	marks the cursor line as the current block (no menu equivalent).

REGISTER