MY HOLIDAY ROMANCE

David Denham

It was hard. It was really hard. But I had made a decision to go for change and I had to stick to it.

Since retirement, I've travelled quite a lot and despite my love for my trusty QL and Gold Card, it was becoming more and more obvious to me that travelling around in a camper van was not really compatible with my QL system. Indeed, it was asking for trouble.

I mean, lugging around a QL monitor, a QL, a Gold Card, a set of disc drives, printer and interface, not to mention power supply and all sorts of other cables did not really make for a restful domestic environment (even though she often pinched time on it to use Quill to write something while we were away), not to mention available space. Never mind the day when I pulled over in a services area and found all the QL gear on the floor in the back having made a rather poor attempt at breaking out and starting a life of its own as I had earlier swerved to avoid a pedestrian in a position no pedestrian should be in.

Scratches and grazes not withstanding, the equipment survived that particular incident. My nerves didn't. So in an effort to prove how much I loved my QL I took the decision that it was high time my wife and I went on our travels and the QL stayed at home.

Thus began the affair with my laptop PC. My wife had been keen to buy one anyway, so in the interests of domestic harmony (roughly translated means "for the sake of my QL") we purchased a used laptop for a good price - if we didn't get on with it we wouldn't have lost much money on it - so that she could have a mobile word processor and I could have a QL emulator running on it.

Not being sure how successful this would be, I decided I wouldn't buy a QPC or even a Qemulator. There is a free option - QLay.

QLay in fact exists in three main incarnations. The original was written by Dutch author Jan Venema in the 1990s, and was produced for DOS, Windows and Linux platforms. Since then, there's been a QLay2, which is an update of the original QLay for Windows. That work was done by Jimmy Montesinos who lives in France. There is a program called QL2K also from Mr Montesinos, but you need to register to use it (I'm not sure if it's free or not), so I started off with QLay 2 and intend to stay with that for now, at least.

The thought of mastering Windows plus mastering an emulator was attractive as a challenge, but not if I wanted to get this up and running before the next trip so that my QL could spend the holiday at home and I could indulge in what might prove to be an axciting little holiday romance with my new computer.

Actually, Windows was already familiar to me to some extent, the main worry was what I would do if I couldn't get the emulator to work. I've used Windows programs plenty of times, but if the emulator didn't work first time, I was not confident of my ability to sort it all out. Could I survive two weeks away from home without a QL? Would I end up for two weeks with only Windows for company? Or would the whole lot be consigned to the bottom drawer (the Denham equivalent of a certain ex-editor's renowned threats to chuck his PC out of windows with a small W). Well, I thought, if it all goes wrong I'll murder the PC, just do the time and come out two weeks later a reformed man.

In practice, it all went fairly smoothly with but a few minor hiccups which were soon overcome, and this is the tale of David Denham's introduction to QL emulators and how he came out the other end safely and without a PC-induced nervous breakdown.

The first step was to get hold of a copy of QLay 2. This was downloaded from Jimmy Montesinos's website at http://www.jadiam.org/QL/QLAY2/

You can download a minimal set of files just to use the emulator (QLAY2.ZIP), or you can download a complete package including source files (QLAY2-Full.ZIP). I chose the former. Anyone who wants to study the sources is encouraged to do so and contribute to its development.

It is well worth making a printout of the instructions on the site. They were apparently written by Simon Goodwin, known to QLers as the author of Speedscreen and Supercharge compiler, and someone who has written extensively on the subject of the QL over the years, including his extensive DIY Toolkit files collection. These notes on QLAY2 include both instructions and a lot of background information. They supplement the instructions which come with the QLay2 package quite well. QLay2's documentation is a bit patchy in fact, many documents based on the original QLay ones, with updates here and there where relevant. While the information is all there, digesting it all can be quite a challenge. For example, the main body of the text of the README.TXT file (the main emulator instructions) consists of Jan Venema's original QLay documents, interspersed with Jimmy's update notes flagged by preceding > marks, like email quotes, which does not make it very readable, although all the information you need is there.

Once downloaded, simply unzip the package to a directory of your choice on your Windows hard disc and create a shortcut to the QLAY2.EXE file, which you will use to start the emulator in future.

The emulator comes with two ROM images, namely a copy of Minerva 1.98 and a ROM image called NFA.ROM (Native File Access - used to store QL files on the Windows hard disc in QLay's own format - more about this later).

You don't have to use Minerva - QLay will happily use any QL ROM image such as a JS or JM ROM. It can also be handy to get hold of a copy of a Toolkit 2 ROM image. I believe it's now freely available from Dilwyn Jones's website, or if you have a QL with an external plug-in Toolkit 2 EPROM, you can save a copy from there with the command SBYTES flp1_tk2_rom,48*1024,16*1024 (i.e. the ROM slot appears 48KB into the memory map and the EPROM is 16KB long).

The emulator has three ROM slots, so you can use a QL ROM image in the BOOT slot, a Toolkit 2 ROM in the next slot and the NFA.ROM in the third slot.

Once you have installed the ROM images in the same directory as the QLAY2.EXE program, you are ready to start the emulator, but there are two other programs you should really install, QLAYT090.ZIP and QLTOOLSQ.ZIP. These are tools programs to aid with the transfer of files between the emulator, floppy discs and the Windows hard drive. At this point you come to realise what is possibly QLay's main weakness, its lack of floppy disc handling. You can't load and save from and to a QL floppy disc. What you have to do is to use the tools programs to transfer files between the emulator's directories on the PC's hard disc and the floppy discs. Even then, life is not that simple, as the QLay directories are essentially Windows hard directories, but with the QL file headers stored in a special directory file called QLAY.DIR. When you save something in the emulator, you have up to 8 WIN drives and another 8 MDV files. MDV files are essentially a directory on the hard disc which pretends to be a microdrive. I never really got to grips with the QLay2 MDV drives, and the WIN drives were more convenient to use for most purposes. But you have to be aware when using the WIN drives that saving or transferring files makes proper use of the QLAY.DIR files or things may not go to plan - if a QL file does not have its header stored in QLAY.DIR, QLAY won't know it exists and any QL executable job header may be lost. Unless QLAY has an entry for a file in its own directory, the file may exist where it ought to be, but will be ignored by QLAY and it will to all intents and purposes be nothing more than a simple Windows data file. And we all know that PCs don't understand QL file headers - they simply get lost and the program can no longer be executed.

This is where the QLAY tools programs come in and they are essential if you intend to transfer files into QLAY or out of QLAY (e.g. to or from QL floppy disc). What the tools programs do is copy the file in the direction indicated, but also try to make sure that the QL file headers are copied correctly too so that QLAY knows about them, and so that the QL will know about them when the disc is read on a normal QL.

QLTOOLSQ v2.7 is a special version of the original QLTOOLS software which was designed for QL/DOS file transfer. QLTOOLSQ is a version modified for use with QLAY, which can copy all files from a QL format disc in the PC's disc drive.

QLAYT is more about moving individual files or lists of files into or out of QLay's directories (normally transfers between Windows and QLay). It can be quite a clumsy affair when you need to send the output of a DOS DIR command to a file, and use that file as a list of files to transfer. Options exist to specify dataspaces and so on, and it can take a while to get used to all the available options. It is worth setting up qltoolsq and qlayt though, as they are the only really effective way of moving files around. What you do is to go into a DOS box in Windows (usually either find Command Prompt in the start menu, or using the RUN command in the Windows start menu, enter RUN CMD to get the command prompt going. Then you type in the name of the program along with whatever parameters the commands need to perform the action required.

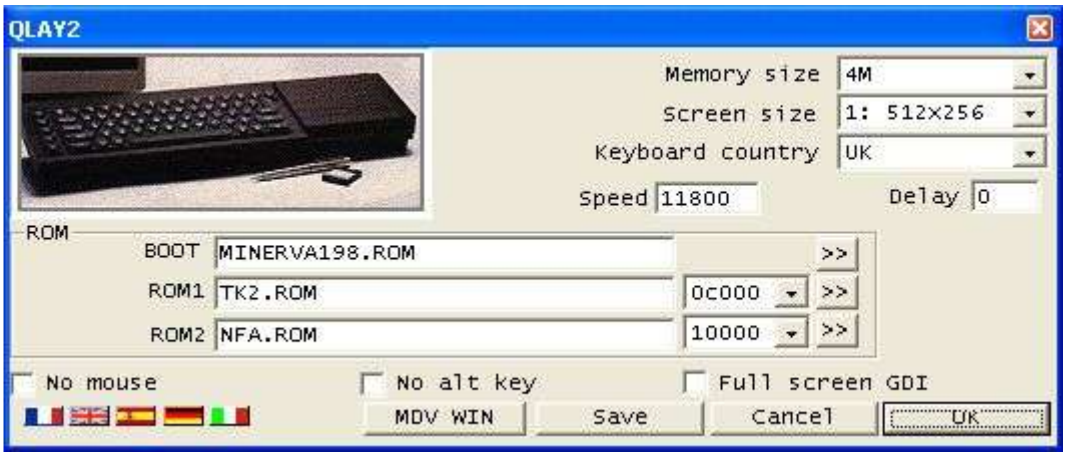
Let's take a brief break here and study what QLay 2 has to offer.

QLay emulates the hardware of a Sinclair QL with 128K, 640K or up to 8MB of RAM. It works with most QL ROM images. As it uses QDOS, it can only offer 512x256 screen resolution since QDOS doesn't know any better, and QLay can't run SMSQ/E. Screen modes 4 and 8 are both available, together with the dual screen capability of the QL, although unsupported by standard QDOS unless you a Minerva able to handle the second screen. QLay can cope with pointer environment. Keyboard emulation via IPC (the QL second processor). A small list of international keyboard variants (British, American, German, French and Italian). A mouse is emulated via memory mapped I/O, allowing the PC's mouse to be used with QLay if pointer environment is used.

The original QLay had IPC sound emulation, but since QLay2 is aimed at more recent versions of Windows such as NT, Windows 2000 and XP, sound is not available in QLay 2 due, the author says, to limitations of NT Kernel Architecture. QLay 2 handles up to 8MB RAM and has a full 68000 emulation, not just 68008 as on the original QL, including all exceptions and trace.

QLay 2 needs a minimum of a Pentium CPU as opposed to a lesser requirement of a 386 CPU for the original QLAY. 9MB of RAM is needed to run the emulator, as opposed to 4MB for the original QLAY, and a minimum of 8MB of hard disc space (4MB if Windows swap files used).

The manual lists some command line switch options for starting QLay, but these seem not to be implemented in QLAY2, instead these are set within a QLAY2.INI file.


QLAY2

Memory size 4M
Screen size 1: 512x256
Keyboard country UK
Speed 11800
Delay 0

ROM
BOOT MINERVA198.ROM
ROM1 TK2.ROM 0c000
ROM2 NFA.ROM 10000

No mouse
No alt key
Full screen GDI

Figure 1 - The startup configuration screen.

There's quite a lot of options here, but once set you probably won't need to change it again.

The list of flags across the bottom selects the language used for the startup screen - just click on the flag for the language required. It defaults to French, but English, Spanish, Italian and German are also available. As you click on the flag, the language displayed in the menus changes immediately.

From the top, the first list lets you select the memory size of the emulator. The usual choice is 1MB, which is fine for most things. You can select values from 128K like an unexpanded QL (useful for very old games which only run on standard 128K systems) right up to a maximum of 8MB, which is twice the memory of even a Super Gold card!

The next box gives the impression of selecting screen resolution, but in fact this is not quite so. Normally, you will select 512x256 like a standard QL, but this occupies only a tiny part of the screen on a modern PC, so QLAY2 rethinks the whole thing and lets you choose to use more than one PC pixel per QL pixel, so that the display is still 512x256 as far as the QL is concerned, but the PC uses more pixels to magnify the picture in effect, making the display look bigger. There's a total of eight sizes to choose from, use whichever suits your requirements and eyesight best.

Next down is the keyboard country required. This lets you specify US, UK, German, French or Italian keyboard layouts. The inclusion of US is especially fortunate for me as my laptop's keyboard seems to conform to US type, I don't know if this is the norm or not.

The next boxes control the speed and delay of the emulator. Speed is especially critical.

The Speed item lets you fine tune the emulator's internal timings. It's not a very easy thing to explain, as it suggests something like how fast the emulator runs, but it's not that easy. Basically, it helps you to set the emulator so that timing-critical things like keyboard repeats and PAUSE delays are correct, so that they correspond to a real QL. Get it wrong, and you may find that the keyboard response becomes way too fast (press a key and you get half a dozen characters instead of the one you expected), or PAUSEs do not wait for the time you expected them to wait. This is how the instructions describe it:

"...it sets the ratio of the emulated QL speed to that of an original QL so that processor-timed operations like key repeat are user friendly on all of the wide range of Windows hosts. The SuperBASIC date and time functions are not affected, as QLay uses the PC's clock for those."

As the manual suggests, watch the flashing cursor and you can often tell if it's not set right as the cursor will flash faster or slower than a QL. Fortunately, there is a simple way to set the speed factor, all you need to do is make a note of the existing Speed value from the setup screen as the emulator starts, then run a small SuperBASIC program called FT_bas for one minute. This program is supplied with the emulator. It will tell you if the speed setting is not right for that particular PC and tell you by how much you need to multiple or divide the existing speed value.

For most PCs, the speed value is usually somewhere in the range of 500 to 1200. The faster the PC, the higher the value. After it has run for one minute, the program tells you how much to change the value. Go back to the startup screen and enter the revised value in the SPEED box. Then click on SAVE to make sure it remembers this new value every time it starts up.

For example, if the existing SPEED value is 500, and it tells you to multiply this value by 1.5, you would enter a value of 750 for SPEED.

FT_bas is a very short BASIC program which simply does 60 one second PAUSEs and compares this with the "real" one minute as determined by the QL clock, which is synchronised to the PC one which is known to be accurate, of course. From this, it can tell how many seconds per minute that the existing SPEED value loses or gains and calculates how much it needs to be changed to correct that difference.

The DELAY value seeks to slow down the emulator to cope with games that run too fast. It doesn't always work, as some older games derive their timing directly from 68008 processor activity, which cannot be emulated 100% accurately, as this is really a PC processor interpreting machine code for another completely different processor. The delay setting is found largely by trial and error - in my case a value of 0 or 1 usually works well, but may vary from PC to PC. It is a question of setting the right compromise between game speed and keyboard response, for example. Set the SPEED value first with FT_bas before you try to set the DELAY value.

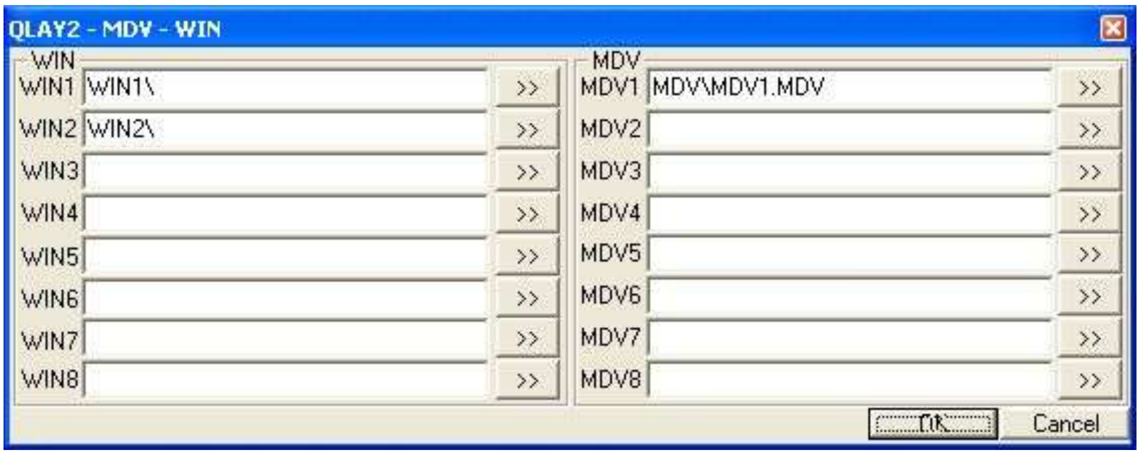
One thing you will find when running QLay2 is that no matter what speed and delay values you set, the load on the PC processor is pretty constant. Minimise QLay 2 and run something else, it will usually be slow and sluggish, depending on the PC processor power of course. QLay must hog a significant percentage of processor time all of the time, but it seems to drop if you click on one of the menus, as though the "QL" is suspended at that point. One way of relieving this load on the PC is to click on the QL menu and select the Exit command and leave the emulator parked on the "Are you sure Yes/No" dialogue, as strnage as that sounds.

The next setup features are the ROM slots. These emulate the ROM addresses on the original QL. The first of the three ROM slots is for the operating system ROM, usually a Minerva or JS ROM, of size 48KB. The second corresponds to the back EPROM slot on a QL, and happily takes the Toolkit 2 16K ROM image. The third slot occupies address space just above the back EPROM slot's 16K address range, at address 65536 in decimal, which I think is used for hardware I/O etc on a real QL. This third slot usually takes the NFA.rom which is a piece of code allowing QLay to use the Windows hard disc to store QL programs, as a WIN file. NFA stands for Native File Access. Note that it does not understand QXL.WIN files at all, although one of the tools programs is supposed to be able to extract files from a QXL.WIN and copy them to the QLay directories. I haven't tried this.

The NO MOUSE option lets you disable the QLay mouse if you find that the Windows pointer and the QLay pointer both show on screen at the same time in different positions. If you use this option, there is no mouse facility when using pointer environment with QLay, although the cursor keys can be used to drive the pointer instead, of course.

NO ALT KEY makes QLay ignore one fo the ALT keys on a PC keyboard so that it can be used for Windows shortcuts. The other ALT key is always available to the emulator anyway irrespective of this setting.

FULL SCREEN GDI scales the QLay display to fill the screen. If you find it annoying that the QLay window only uses a small part of the PC's larger screen display, use this option to allow the QLay display to expand to fill the screen. Resolution is still 512x256 - this just makes the PC use several Windows pixels per QL pixel to scale it to fit the PC screen. Depending on how good your PC is at scaling graphics and what PC screen resolution you use, it may produce good or very ugly results because you may find that the scaling needs to use, say, 2 pixels for some QL lines and 3 pixels for other QL lines to scale it correctly to fit the whole screen. This means some horizontal lines are thin, some thicker, and it makes for a very ugly result in extreme cases, especially with text screens. Again, largely a question of trial and error to get the best compromise possible. If you can't make the full screen GDI work to your satisfaction, try the various SCREEN SIZE options instead, as these can use consistent numbers of PC pixels to QL pixels to make the display look normal even if they don't quite fill all of the screen.


QLAY2 - MDV - WIN

WIN			MDV		
WIN1	WIN1\	>>	MDV1	MDV\MDV1.MDV	>>
WIN2	WIN2\	>>	MDV2		>>
WIN3		>>	MDV3		>>
WIN4		>>	MDV4		>>
WIN5		>>	MDV5		>>
WIN6		>>	MDV6		>>
WIN7		>>	MDV7		>>
WIN8		>>	MDV8		>>

Figure 2 – the MDV/WIN setup screen

Finally, we come to the MDV/WIN button. Click on this and a complex looking screen appears which lets you define where QLay2's WIN and MDV drives live on the PC hard disc. Think of the QLay WIN drives as being up to 8 separate hard discs as far as QLay is concerned. Each can store up to 159 files, giving you over 1200 files in total, more than enough for most people I'd have thought. The MDV drives are very small (255 sectors, which is the maximum theoretical capacity of a microdrive cartridge, though most cartridges are about 200 to 220 sectors) drives, intended to emulate microdrives on the original QL. It can be useful to transfer older games into one of these to see if you can get them to work on QLay MDV if you can't work out how to configure the game to run from a WIN drive.

Although it can only handle 8 drives at a time, you can set up as many as you like and switch between them using the setup screen if need be. So you could set up one copy of WIN1_ to contains your work information, another for your home use, another for your partner and so on.

Hard discs for a QL are a subject new to me, though I've made use of directories and so on on a floppy disc drive with a Gold Card. But QLay doesn't use directories - it only has what is called a Level 1 filing system, no MAKE_DIR command, so you can't go putting all your programs into separate directories. So think of each WIN drive as a large capacity floppy disc drive called WIN1_, WIN2_ and so on. In a way, the limit of 160 or so files per WIN drive is a good thing. Imagine trying to find your files amid thousands of other files!

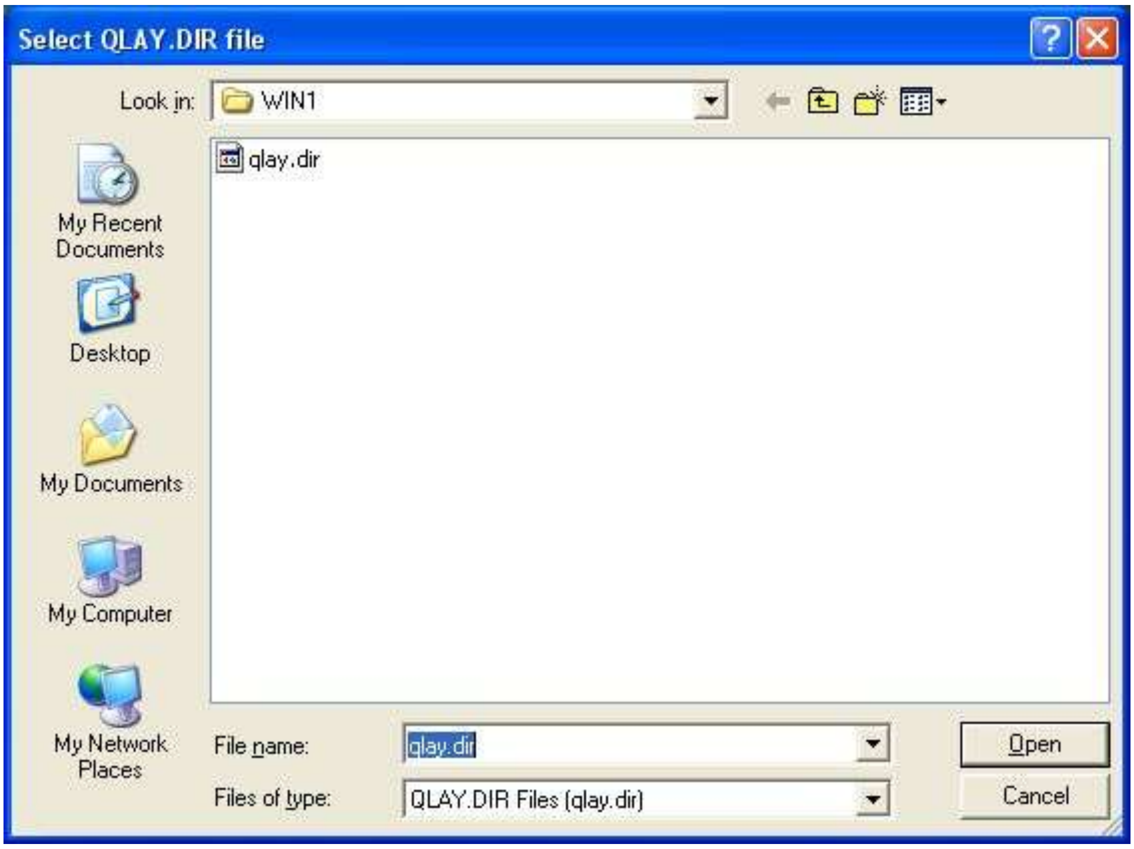
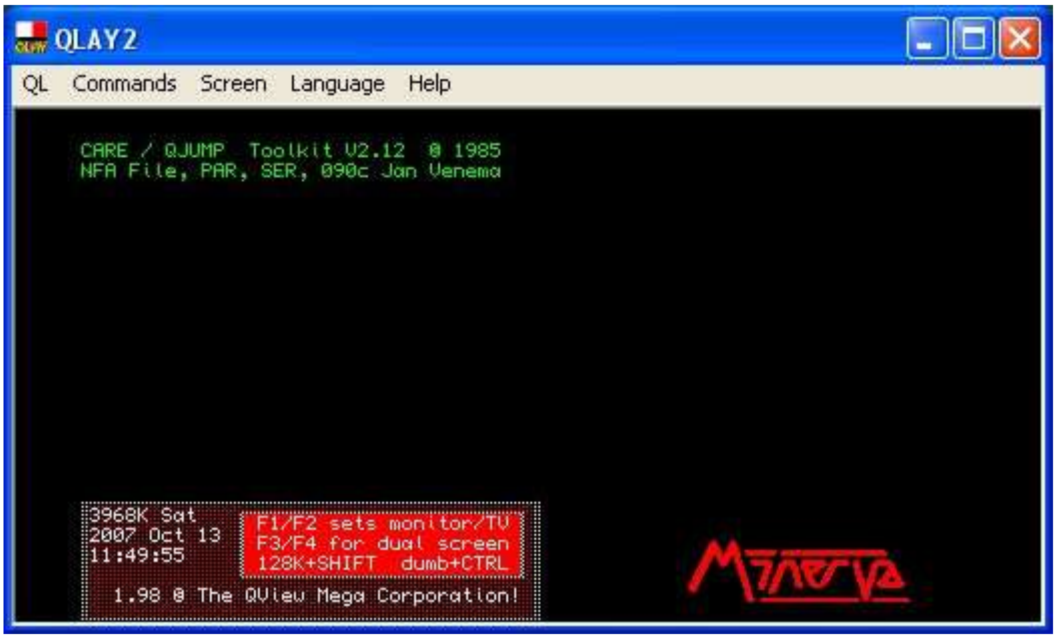


Figure 3 - browsing to find a QLay WIN drive.

This actually finds folders with a QLAY.DIR file, a file which contains the headers for the QL files stored, so it does manage to give the impression of having to create something which already exists, but don't be too worried about it, it's quite easy. If you get stuck, you can always go into Windows and copy a QLay.DIR file from one place to another to give you a head start.

I found it most convenient to put these WIN drives in the same PC directory as QLay itself. So, I put QLay2 into a folder called QLay on the C: drive. I created folders called WIN1, WIN2, WIN3, WIN4 and so on inside the QLay directory from Windows, then used the browse buttons in the MDV-WIN screen (the >> buttons to the right of each drive name) to find these folders and point QLay to them. Although I did set up 8 MDV drives as well, I found them less useful and haven't really used them since then. Once all the drives are set up, click on the OK button to return to the main setup screen, then click on the SAVE button to store these settings for future use. Finally click on the OK button in the main setup screen and the emulator starts.

The familiar QL screen is shown and it prompts you to press F1 or F2 for Monitor or TV windows mode as a standard QL would. If using the Minerva ROM, the emulator may get fed up of waiting for you after a while and start up of its own accord. This was a feature built into Minerva in the days when bulletin boards rather than the internet were the norm, so when a power cut occurred, or the QL reset for some reason, the QL could start itself up automatically and run its boot program to restore normal service again when the power was restored.



```
CARE / QJUMP  Toolkit V2.12  © 1985
NFA File, PAR, SER, 090c Jan Venema

3968K Sat      F1/F2 sets monitor/TV
2007 Oct 13    F3/F4 for dual screen
11:49:55       128K+SHIFT  dumb+CTRL
  1.98 © The QView Mega Corporation!
```

Figure 4 - Minerva starting up.

OK, we now have an emulator set up and ready to go and up to 8 drives with nothing in them, so where do we go from here?

If you haven't already done so, lrun the FT_bas program to make sure the emulator speed is correct.

The next stage will be to transfer some files from the real QL using a QDOS floppy disc.

To do this, we need the qltoolsq program supplied with QLay2. This is a DOS program which we run from Windows. Make sure that the folder which contains QLay2's WIN drive contains copies of the PC programs called QLAYT.EXE and QLTOOLS.EXE.

QLAYT.EXE helps with transferring files between Windows and QLay2.

QLTOOLS.EXE helps with handling QDOS formatted floppy discs and transferring files between the QL floppy disc and the QLay WIN drives.

The simplest way of running these is to start a DOS command box in Windows. In the Windows START menu, go to the Accessories sub-menu and click on Command Prompt (not sure if this varies between different versions of Windows). Alternatively, use the Run command from the START menu to run "cmd" (without the quotes). We need to change to the QLay2 directory:

```
cd qlay
```

ensure there is a QL floppy disc in the PC's disc drive, with some files on it. Now enter the command:

```
qltools a: -d
```

This will list the files on the QL floppy disc. To transfer these files to QLay and update QLay's directory (contained in a file called QLAY.DIR) enter this command:

```
qltools a: -q
```

Qltools has many other commands. This one is probably the most useful, as it transfers all files from the disc in one go, and creates the entries in the QLay2 directory automatically. Copying files from Windows or from the QLay directories to the QL floppy disc is also possible, using "qltools -w filename" (without the quotes) which copies the file specified to the floppy disc.

Qltools.exe has its own instructions files, but you can get it to display a help screen by simply starting it with nothing after the qltools command.

QLayt.exe is a program to copy, update, delete etc files between Windows and QLay2. Using something like Windows Explorer it is easy to copy or drag and drop files into the folder which is the QLay2 WIN file. But just copy it there and QLay does not see it, because it hasn't yet been added to the directory list in QLAY.DIR. This is where QLAYT.EXE comes in.

There are two ways of using this DOS program. The first is to act directly on files, the other is to create a list of files and let qlayt work on the list instead.

If your QLay WIN drives do not contain a file called QLAY.DIR, use QLAYT to create one while adding files.

Firstly, send a list fo files to be transferred to a PC file called list.txt - we can use the DOS command DIR with a '>' command to send the output to a file instead of to the screen:

```
DIR /b >list.txt
```

The /b forces the DOS command DIR to just list the simple filenames without details of length etc to make the list easier to hande.

Then we use the following command to create a new QLAY.DIR and add the files listed in "list.txt" to QLay's directory:

```
qlayt -c list.txt
```

The '-c' forces qlayt to create a new directory file if none exists already. If you just want to add (append) the files list to an existing QLay directory without creating a new one, use -a instead:

```
qlayt -a list.txt
```

After creating or adding files to QLay's directory, you may feel more comfortable checking it before you proceed. To do this, use qlayt -l (that is, a lower case letter L not the number 1):

```
qlayt -l
```

This will show if the file is listed in QLAY.DIR, which will ensure QLay2 sees the file. If you wish to, you can even send this list to a file, e.g. if you are a neat and organised person who likes to keep accurate records of files, or simply to create a list to help you with copying files from the emulator to Windows:

```
qlayt -l >qlaylist.txt
```

Using lists like this is a bit clumsy but OK for a long list of files because you don't have to type in a qlayt command for each and every file. But if you want to add just one file from Windows to QLay, it is easier to use the include' command option to specify that you wish to append just one file to QLay's directory. In Windows, copy that file to the folder holding QLay's directory then tell qlayt to inform QLay that the file is now added to its directory. Suppose it's a file called README.TXT:

```
qlayt -i README.TXT
```

Basically, as QLAY.DIR holds the QL file headers (details of length, file type etc) for the files which live in QLay's WIN folder, all it does is to add the 64 byte file header for README.TXT into QLAY.DIR so that QLay2 becomes aware it exists. QLay2 should not be running when you use QLTOOLS or QLAYT in this way, it only becomes aware of changes when it is first started.

If the file to be added is a QL executable job, things are a bit more complicated since Windows doesn't understand QL file headers, so as far as QLay would be concerned, a QL program is just another data file, not a program. Suppose we have a QL program called TEST_TASK which has been copied to QLay's folder and we wish to make QLay aware the program exists and is in fact an executable QL program. We have to make sure that QLay2 has a suitable file header which includes the dataspace value needed by that program when it runs on a QL. Some programs, such as those compiled with C68, contain the dataspace value in something called the "XTcc" field, embedded in the file itself. QLAYT can scan for this and try to extract the dataspace for the program:

```
qlayt -i test_task -X
```

The -X, the letter after the - ("switches") must be in the correct case, tells QLAYT to try to find the dataspace from an XTcc field in the program itself. This doesn't always work, so you may have to resort to plan B - specify the dataspace yourself:

```
qlayt -i test_task -d 2048
```

the "-d 2048" indicates that the directory entry for this file should be for an executable file with a dataspace value of 2048 bytes (it should be a decimal number).

If you are unsure of the dataspace value required for a given program, you can check it on your QL before transferring the file to QLay. If you have Turbo compiler, use its dataspace_task program to tell you the dataspace, or use the FDAT function of Toolkit 2:

```
PRINT FDAT(_test_task)
```

If you are completely unsure, you can often get away with giving a safe, high value. Programs don't mind being given too much dataspace as a rule, although it is wasteful of free memory.

If all else fails, you can transfer the "program" to QLay as a data file, then use ALCHP/LBYTES/SEXEC commands in QLay2 to convert to an executable. Another method which is harder work is to zip up a program on a QL using the QL ZIP program and transfer the zipped program to QLay as a data file then use QL Unzip on QLay to unzip the program. Being held in a QL _zip file protects the job header and dataspace values.

The QLAYT manual lists several ways of transferring files from a QXL.WIN to QLay directories, although I didn't have a QXL.WIN to hand to test this. I suppose you need to be aware that QLay WIN drives can only have a maximum of 159 files each, and that QLay2 doesn't handle level 2 directories. Use the -E option to Extract files from a QXL.WIN in this way. First, create a list, then add the content of the list to QLay2's directory:

```
qlayt -E qxl.win >list.txt
qlayt -c list.txt -q qlay.dir
```

The first line extracts a list of files from the QXL.WIN. Using a list has the advantage that you can load the list into Windows Notepad editor and manually alter the list if required. For example, if the QXL.WIN contains thousands of files you can simply delete the ones not needed to make the list much shorter. The second command creates a new qlay.dir (the -c option) and puts the files listed in list.txt into it, then the -q option ensures that the new directory file created is called QLAY.DIR. I should emphasise, I didn't have a QXL.WIN to hand and the above example is based on my assumptions and understanding of the manual and I am probably completely wrong (assumption is the mother of all foul-ups, of course!)

A simpler method is probably to extract and add direct with something like:

```
qlayt -E qxl.win -q qlay.dir
```

If you wish to transfer files to a MDV directory instead, use -C (note: upper case C this time!):

```
qlayt -C list.txt
```

This will create an MDV image file from the files listed in list.txt.

You can copy files from an MDV slot to a WIN directory with the -W option:

```
qlayt -W mdvfile
```

Most of these commands use the current QLAY.DIR directory. Adding a "-q name" part to the command will tell it to use another directory file instead if required.

In addition to the example commands given above, there are commands to remove filenames from QLay2 directories, convert files between QDOS and Windows formats and vice versa, update header information if a file in a QLay2 directory has been altered by Windows and even to insert a random sector number on MDV files (I assume that this is to cope with MDV programs which are copy protected with a mechanism which detects a particular random number placed on a MDV tape when it was formatted).

QLAYT can do a surprising number of things and all the possible commands and permutations would probably warrant a complete book, not just an article, to itself.

QLay 2 has a Windows menu system replacing combination keypresses and some command line options in older versions of QLay. For example, in older versions of QLay, you could press CTRL ALT SHIFT and R to force a reset, like pressing the rest button on a QL. In this version, there is a menu bar across the top of the emulator display and this includes a reset command.

Back to QLay2 itself and how it works in practice. Compatibility with existing software is quite good, once you have successfully transferred them to QLay2 directories and configured everything to run from WIN. What is less good is the sluggishness of the PC when trying to run something else, and how QLay2's screen update can become quite sluggish or even stop altogether when running intensively, the only way to restore the screen when this happens seems to be to click outside of QLay and then back into QLay, and I'm not quite sure why unless it's something to do with suspending QLay processing and giving I/O a chance to catch up.

Things which don't work at all include sound (no BEEP, since the Windows 2000 derivatives of Windows such as XP don't support the required facilities in the kernel architecture), and floppy disc access from within the emulator (you have to use the tools programs). I am not sure about PAR and SER ports, as I couldn't get PAR to work and I have nothing connected to a SER port to test it. The failure to get PAR printing working seemed rather strange as the startup screen lists "NFA file, PAR, SER..." in the ROM identification line at the top of the QL screen (see figure 4) but try as I might to print to "PAR" or "PAR_" I couldn't get it to work.

Things which do work and work well include SuperBASIC and the ability to run existing software. You don't have to cope with the new facilities of SMSQ/E which QPC would force upon you, for example. QLay 2 cannot run SMSQ/E and is unashamedly good old QDOS, which will probably be important to those who only use the emulator from time to time to run a few favourite programs, or who may be using an emulator for nostalgic reasons - their first "QL" for years. In my case, I only use QLay2 when I'm away from home - I revert to using the QL at home.

Was it worth the installation? Given that it was free, and the bits which work work very well indeed, QLay 2 is a simple and perfectly adequate way to run QDOS on a PC. It has filled my need for a system to let me run QL software on a second and more portable machine in the form of our laptop. I've felt a bit frustrated by the lack of direct floppy disc handling, but once I got used to using the hard disc via the WIN drives, I found I didn't need the floppy discs from day to day, so I only needed to resort to the tools programs every once in a while if I wanted to copy something to carry over to the main QL via floppy disc. It was handy being able to transfer files such as documents between Windows and QDOS so simply, using the QLAYT program. QLay2 ran most of the QDOS programs I threw at it, including many of those needing pointer environment. It can easily match the speed of a standard QL on most fairly modern PCs, even exceed it in some cases. Usefully, you can also slow it down for playing fast games, a great help to someone of my age of course!

I wouldn't have paid big money for this emulator, but given that it's free I found it to be quite adequate for the things I needed of it - and I would pay it even higher praise if parallel port printing could be amde to work and it could read QL floppy disks directly in the emulator.

So, I reckon I'll be keeping in touch with my cute little holiday romance in the future!

And here is how you can share the romance - download QLay2 from this website:

http://www.jadiam.org/QL/QLAY2/