The file system

Many games and applications written on the Amiga or Atari ST computer accessed the file system, to load images or sounds, or access sequential or random access files. It was therefore mandatory to emulate in some way the file system of those machine so that the games and application can be immediately compilable on modern machines with AOZ.

The AOZ compiler contains code that emulate the drive and directory structure of an Amiga machine. This structure is also compatible with the Atari ST file system.

How does it work?

- A folder named 'filesystem' lays in your application folder (please see previous chapter)
- Every directory contained in this 'filesystem' folder defines a 'drive'
- Files and folders contained in each virtual drive will be integrated in your compiled application and available from within the program for loading

Example of definition of such directories

```
filesystem
    DHO
        libs
            amos.library
            windows.library
            sprite.library
        my_folder
            myapp.amos
            readme.txt
        clickme.app
    DF1
        game.exe
        me.iff
        bob.iff
        sound.iff
        Data
            info.txt
            scores.txt
```

If you compile your application and include in the code a Dir "DH0:" command, you will see the result:

Directory of DH0:

- libs
- my_folder
  clickme.app   1000 bytes

Dir "DH0:my_folder"
Directory of DHO:my_folder:
myapp.amos   12814 bytes
readme.txt   1481 bytes

A complete emulation

AOZ directory and file functions work exactly the same as on the original Amiga or Atari ST. Commands such as Dir, Dir$, load, Bload, Load Iff, Open Out, Close, Print # work exactly the same as on the original machine.

The goal is that original AMOS application believes it is still running on an Amiga so that compilation is right at first time and you do not have to change the code. So be sure, if you port your application from original AMOS or STOS to AOZ, to eventually copy the necessary files in the 'filesystem' folder.

You can of course create new directories, delete and save files (see later).

The 'application' drive

Even if it is not defined in the 'filesystem' folder, AOZ defines a default drive that will always be present for any application, called 'application:'. This drive is empty (yet present) if it is not defined in the 'filesystem' folder.

The default directory of any AOZ application at start is "application:". Therefore, a simple Dir command at the start of your application will not generate any error (but display nothing):

```
Dir
Directory of application:

```

```
Print Dir$
application:
```

The 'application:' drive is a very easy way to append data to your application. Simply create the directory as a sub-directory of the 'filesystem' directory and copy anything you want in it. But remember, all those files have to be encoded and attached to the source code of your application, and will increase its size.

Saving files

Saving files from a browser is a much more difficult problem than reading it, due to the build in protections to safegard the machine of the user. The current version of AOZ does its best to overcome those limitations by using a system called 'LocalStorage' that is available in every modern browser.

'LocalStorage' is a persistent disc space that browser makes available for every web-site you visit. It is a form of 'enhanced cookie system'.

When you save a file from AOZ (either by using Bsave, Open Out, Open Random or Save), the file is saved in the LocalStorage area. The next time you start the same application with the same URL, the file will still be there.

The process is entierely transparent to you, the programmer. Even if the 'data' of the file are saved in the LocalStorage area, the file will appear from your AOZ application integrated in the file system, and its path will be the path you saved it to.

Examples:

```
Reserve As Work 10, 10000
For X = 0 To 10000
    Poke Start( 10 ) + X, Rnd( 255 )
Next
Bsave "mydata.bin", Start( 10 ) To Start( 10 ) + Length( 10 )

Dir
Directory of application:
mydata.bin                  10000 bytes

Mkdir "new_directory"
Dir$ = "application:new_directory"
Bsave "mydata2.bin", Start( 10 ) To Start( 10 ) + 5000

Dir
Directory of application:new_directory
mydata2.bin                 5000 bytes
```

The size of the LocalStorage area depends on the browser used to run the AOZ application, and can be found with the Dfree or Disc Info functions. Chrome reserves 5 MB for this space and Firefox 10 MB.

This might sound small for today's point of view, but is largely enough for Amiga or Atari ST applications...

LocalStorage is only the first implementation of saving data, new methods will be added in future versions:

- direct save on the user's machine when the application is wrapped as a native executable
- saving on the server where the application is hosted, by the mean of a small websocket server that I intend to program