Structure of an AOZ application

In the early ages of computing, disc space was scarced, many people did not have a hard-drive and had to save their projects on slow floppy-discs. This is the reasons why the original AMOS saved the applications as one single file, packing all the graphics, sounds and code into one single, simple to manipulate file.

Things have changed, and today everyone has a hard-drive. Modern editing tools deal with single files (text, images, sounds, videos etc.) and the original concept of 'all in one file' would not have been practical for AOZ.

AOZ applications are therefore contained in a folder, and each one of the component of the application are separate files, organised in a logicial directory structure.

In a future version, you will be able to directly compile .amos files. The compiler will extract all the elements of the application (images, sounds, source code etc.) and create the resulting directory structure for you. As a result, it will transform your old games that were only editable on the Amiga into real modern projects editable with modern tools on nowadays machines.

The directories

The basic structure of an AOZ application is as follow:

```
Root_Directory (name of the application)
    resources
        images
            1.png
            2.png
        icons
            1.png
            2.png
            3.png
        musics
            1.mus
            2.mus
        AMAL
            1.amal
            2.amal
            3.amal
        samples
            1.wav
            2.mp3
            3.ogg
        bankname_banknumber (example menu_5, picpac_10, tracker_7, data_11....)
            number.extension
            number.extension
            number.extension

    filesystem
        application
            folder
                ...files
            folder
```

```
                ...files
            ...files
        DHO
            folder1
                subfolder
                    file1.ext
                    file1.ext
                file1.ext
                file1.ext
            folder2
                subfolder
                    ...
                subfolder
                    ...
                ...
    manifest.hjson
    main.amos
```

The files

manifest.hjon

This file contains the compiler and runtime properties of the application. A detailed explanation of this file is done in the next chapter.

main.aoz

This file contains the source code of the application, in normal UTF-8 format. Any source code editor can be used to modify it (such as Visual Stutio Code, Notepad++, Jetbrains line of products etc.). The compiler understands all cariage return formats and will compile applications saved on Linux, Windows or MacOS.

resources folder

This directory contains the 'memory banks' of the application, the content of which saved as individual files.

images folder

This directory represents memory bank #1, the 'Sprites' bank. The images of the bank should be PNG or JPG (formats that browsers can recognize). The number of each element is indicated in the filename itself, for example the file '1.png' will be available within the AOZ program as sprite #1 in the bank, and the following instruction will display it as a bob:

```
bob 1, 100, 100, 1
```

Also note (not yet completely debugged in the current version), that you can save pictures with real filenames, for example 'mysprite.png' and that the bob and sprites command will allow you to call the images by their original names (example: `bob 1, 100, 100, "mysprite.png"`)... The goal of this enhancement is to make AOZ program clearer and simpler to make when you have a large amount of sprites.

icons folder

This folder contains the elements of memory bank #2 in original AMOS, the 'icons' bank. As for the sprites, each icon is a numbered image file. Files in the icon bank can only be png, and any other file extension will be ignored by the compiler.

musics folder

This directory contains the elements of bank #3 of the original AMOS, the "music" bank. musics are not yet implemented in the compiler.

AMAL folder

This directory contains the elements of the AMAL bank, bank #4 of the original AMOS application.

samples folder

This directory contains the elements of the Sample bank, bank #5 of the original AMOS application. As for sprites and icons, each sound should be a numbered file. Format supported will be the ones allowed in a browser, WAV, MP3 and OGG.

Other formats of banks

The five first banks were 'reserved' memory banks in AMOS, but you could also define your own banks and save them with your application. AOZ Studio improves this functionality, and allows you to define any kind of bank, and include any file in it. The principle is simple:

A bank is defined by the name of the folder: 'bankname_banknumber'. For example, a folder named 'data_8' will be listed within the aoz application as a 'data' bank, and will bear the number 8. The underscore indicates the separation between the name of the bank and its number. The compiler will generate an error if there are more than one underscore in the folder name, or if the characters after the underscore and not parsable into a integer number. It will also generate an error if the number is below 5 or already used in another folder.

The CONTENT of the bank is build from the files found within it. When compiling, AOZ scans each directory and lists the files, then sorts them alphabetically. It then includes the binary content of the file into the bank. Each element will be available with a new function (yet to implement), = Start Bankitem(bank, element) and = Length Bankitem(bank, element) that will be complementaty to the Start and Length functions.

Note that if the bank only contains one element, the Start and Length function will be 100% compatible with the original AMOS.

Some names of banks are reserved, and can only contain specific data related to their use:

- "picpac" : contains images packed with original AMOS picture compactor
- "menu" : contains menus to be display with AMOS menu instructions
- "resource" : contains the definition of dialog and buttons for the AMOS interface instructions (not implemented in AOZ)
- "tracker" : contains .mod files and other compatible formats

filesystem folder

The filesystem folder allows AOZ compiler to emulate the Amiga filesystem within your application, and ensures compatibilities with applications that loaded elements from the disc.

It also represent a very simple way to append data to your application.

How does it work?

- The root filesystem folder contains subfolders, each sub-folder represents a 'drive' in the Amiga sense of the term. Files are forbidden at this level, and will generate compilation errors. The name of each folder defines the name of the drive when the application runs. Examples of names:
    - "DHO" : your application will have a "DHO:' drive mounted when it runs
    - "DF1" : your application will have a "DF1:" drive mounted when it runs
    - "application" : even if it is not defined, an "application:" drive is available to each application, and this drive is the default path of the application. Creating the folder in the "filesystem" folder allows you to populate it with files and folders.
- Files and folders located within each "drive" directory will be reflected in your application as if they were present on the real disc. AOZ instructions work as if they were using the real original filesystem. You can list the files with the Dir command, you can change the directory with the PATH$ reserved variable, got o the parent directory with the Parent instruction etc. Load, Bload, Load Iff are of course also supported.

Please refer to the "filesystem" chapter for more information.