Bash Scripts for Audio-Visual Preservation

Reto Kromer • AV Preservation by reto.ch

IASA Conference & ICTMD Forum
Istanbul, Türkiye
from 11 to 14 September 2023

AV Preservation by reto.ch

If it's not awesome, we don't do it.

A stencil-coloured print of
Création de la Serpentine *by Segundo de Chomón,*
Pathé Frères, France 1908
(Courtesy by Österreichisches Filmmuseum, the Austrian Film Museum)

We are a highly skilled moving image conservation and restoration company, partly operating at the Lichtspiel in Bern, Switzerland. Our team provides worldwide comprehensive services that encompass the whole range of audio-visual preservation, offering Swiss quality at a competitive price. We have a particular commitment to the dissemination of low-technology and low-energy audio-visual conservation and restoration techniques. We support open-source projects which we believe are beneficial for the archival community. We are running the Swiss Film and Video Directory.

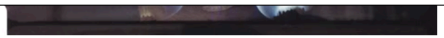
Frequent and Rarely Asked Questions on Audio-Visual Preservation Issues

Digital is cool. And analogue is hot.

Bayer misalignment for … pain and fun.

Our goal is to discuss myths and science on audio-visual conservation and restoration issues, and to distillate solid founded resources for the field.

Table of Contents



Bayer misalignment for … pain and fun.

Our goal is to discuss myths and science on audio-visual conservation and restoration issues, and to distillate solid founded resources for the field.

Table of Contents

2022-11-26

Bash Scripts for Audio-Visual Preservation

Bash AVpres is a set of Bash scripts for audio-visual preservation, used on a daily base at our conservation and restoration lab, AV Preservation by reto.ch. These small programs are designed for both t be used individually and to be chained each other as needed. Bug reports are very welcome, as usual.

Table of Contents

Elements of the Set

Checksum manifests

- make_manifest – make a checksum manifest
- verify_manifest – verify a checksum manifest
- update_manifest – update a checksum manifest when files are added or deleted
- name_hash – add, verify or remove a checksum as filename's suffix
- make_framemd5 – make a frame MD5 checksum manifest
- verify_framemd5 – verify a frame MD5 checksum manifest

Files

- make_ffv1 – make a FFV1 file

Elements of the Set

Checksum manifests

- make_manifest – make a checksum manifest
- verify_manifest – verify a checksum manifest
- update_manifest – update a checksum manifest when files are added or deleted
- name_hash – add, verify or remove a checksum as filename's suffix
- make_framemd5 – make a frame MD5 checksum manifest
- verify_framemd5 – verify a frame MD5 checksum manifest

Files

- make_ffv1 – make a FFV1 file
- verify_ffv1 – verify a FFV1 file
- make_h264 – make a H.264 file
- make_prores – make a ProRes file
- metadata_csv – recursively create a CSV file with metadata of all AV files in a folder
- missing_files – find missing files in a folder of sequentially numbered files

BagIt archives

- make_bagit – make a BagIt archive
- verify_bagit – verify a BagIt archive
- update_bagit – update a BagIt archive when files are added or deleted [beta testing]
- modify_bagit – modify the BagIt version
- undo_bagit – undo a BagIt archive

FFmpeg tools

- ffengine_presets – list, install, delete or print *FFCommand Engine* presets (macOS and Windows)
- ffmpeg_head – install, patch or delete FFmpeg HEAD

Documentation

Documentation

All the scripts come with a short embedded help message and a manual page ("man"). Running `bash_avpres` shows the list of the installed commands:

```
$ bash_avpres

Bash AVpres 2023-03-26 provides the following commands:
  make_bagit               undo_bagit               metadata_csv
  make_ffv1                                         missing_files
  make_framemd5            update_bagit
  make_h264                update_manifest          name_hash
  make_manifest
  make_prores              verify_bagit             bash_avpres
                           verify_ffv1              ffengine_presets
  modify_bagit             verify_framemd5          ffmpeg_head
                           verify_manifest          nano_config

$
```

Installation

The scripts can be run from everywhere, without any specific installation. Of course, they have to be executable; if they aren't, run for example `chmod +x bash_avpres` or, when administrator privileges are required, `sudo chmod +x bash_avpres`.

In addition, two possibilities are provided for a regular installation at the root: via a Homebrew formula ar via a Makefile. And we advise to use either one of these.

Installation

The scripts can be run from everywhere, without any specific installation. Of course, they have to be executable; if they aren't, run for example `chmod +x bash_avpres` or, when administrator privileges ar required, `sudo chmod +x bash_avpres`.

In addition, two possibilities are provided for a regular installation at the root: via a Homebrew formula a via a Makefile. And we advise to use either one of these.

HOMEBREW

The installation via Homebrew works fine not only on Linux and Mac, but also on Windows running Terminal or Subsystem for Linux. Run the following two commands in the Terminal:

```
brew tap avpres/formulae
brew install bash-avpres
```

MAKEFILE

As usual, directions are recalled in the `README.txt` file. Run the following three classic commands in the Terminal:

```
cd bash-avpres-2023-03-26
./configure
make install
```

Compatibility

The **Bash AVpres** scripts have been used successfully on various modern x86_64 and AArch64 architectures, running under the following operating systems:

- **Linux:** Debian 12.0, 11.7 and 10.13; Ubuntu 23.04, 22.04.3 LTS and 20.04.6 LTS; Slackware 15.0

Compatibility

The **Bash AVpres** scripts have been used successfully on various modern x86_64 and AArch64 architectures, running under the following operating systems:

- **Linux:** Debian 12.0, 11.7 and 10.13; Ubuntu 23.04, 22.04.3 LTS and 20.04.6 LTS; Slackware 15.0
- **Mac:** macOS 13.5.1, 12.6.7 and 11.7.8
- **Windows:** 11 version 22H2 and 10 version 22H2, running Terminal or Subsystem for Linux

Almost all the scripts have been programmed to run also on the old Bash version 3.2 (released on 2006-10-11), which sadly still comes with the Apple computers, but one runs better on Bash version 4.3 (released on 2014-02-26). However, we strongly advise to install the current Bash version 5.2 (released on 2022-09-26) also on computers running under macOS. Note that we didn't check any compatibility with versions older than 2.04 (released on 2000-03-21).

Source Code

The source code of the **Bash AVpres** package is available on our website as a ".tar.gz" file, a TAR archive which was compressed with `gzip`. Of course, it includes a Change Log file.

Acknowledgments

Reto Kromer wishes to acknowledge the inspiration given and the help provided by:

- Frédéric Noyer
- Joshua Levy
- Joshua Ng
- Michal Cohen

(in alphabetical order of the given name).

10-11), which sadly still comes with the Apple computers, but one runs better on Bash version 4.3 (released on 2014-02-26). However, we strongly advise to install the current Bash version 5.2 (released on 2022-09-26) also on computers running under macOS. Note that we didn't check any compatibility with versions older than 2.04 (released on 2000-03-21).

Source Code

The source code of the **Bash AVpres** package is available on our website as a ".tar.gz" file, a TAR archive which was compressed with `gzip`. Of course, it includes a Change Log file.

Acknowledgments

Reto Kromer wishes to acknowledge the inspiration given and the help provided by:

- Frédéric Noyer
- Joshua Levy
- Joshua Ng
- Michal Cohen

(in alphabetical order of the given name).

Copyright, License and Disclaimer

2023-08-22

Summary

- history
- current set of scripts
- other approaches

Programming Languages

C

- codec, image and sound processing

Perl

- network, website, database, administration

Bash

- scripting

Help

- embedded help message
- comprehensive manual page ("man")

Parameters

- passed when called
- configuration file
- hard-coded default values

Current Set

- checksum manifests
- audio-visual file generation
- technical metadata extraction
- BagIt File Packaging Format
- FFmpeg tools

Checksums

- various algorithms
- manifest vs. filename

Algorithm

cryptographic

- MD5
- SHA-1
- SHA-256
- SHA-512

non-cryptographic

- CRC-32
- xxHash 32
- xxHash 64
- xxHash 128

On the Bright Side of Data Migrations

Reto Kromer, AV Preservation by reto.ch, Switzerland

Let's be very clear from the very beginning: I do not consider data migration a good thing at all for the archive community. On the contrary, it costs a lot of time, money, and effort to be achieved accurately. But it cannot be avoided. I will discuss here how data migrations can be used efficiently for modifying, where necessary, the archive's containers, codecs, data and metadata. During the two dozen of data migrations we have carried out for ourselves and our clients, we could actually fix errors in the structure and metadata of the archive, and also we could replace obsolete or endangered formats with current ones. This allows us to change or adjust the strategy when needed. We could update the data and therefore realise maintenance of the digital archive.

1. In the Jungle of File Formats

When I announced that I was going to present on the bright sides of data migration,[1] my colleagues replied to me that it will be a very short speech…five or ten seconds at best! And, indeed, data migration is mostly a bad thing, also because it can be imposed by vendors on archives (when archives have service contracts for software or hardware products) and it can cost a lot of time and money that archives often lack. Inaction is almost never a good choice in the digital domain. Often, it is better to choose an intermediate step, one that can be improved or modified later. Nevertheless, my presentations—and consequently this article—do focus on how a data migration can be used in a positive way, for example to modify where needed the archive's containers and codecs. And I do apologise that my text has a strong personal tone.

Include Checksum into Filename

rather than two files

- Title_ffv1.mkv
- Title_ffv1_mkv_xxh128.txt

use only one

- Title_ffv1_9d5084b5b0a08d5022b39e0e75241d12.mkv


```
[10:23:55]imac01@iMac01:~/Desktop$ name_hash
Help:
  name_hash -h
[10:25:09]imac01@iMac01:~/Desktop$
```

```
[10:23:55]imac01@iMac01:~/Desktop$ name_hash
Help:
  name_hash -h
[10:25:09]imac01@iMac01:~/Desktop$ name_hash -h

Usage:
  name_hash (-a|-v|-r) <input_path>
  name_hash -h | -x
Options:
  -a  add checksum to filename
  -v  verify if file's content and checksum matches
  -r  remove checksum from filename
  -h  this help
  -x  advanced options with their default arguments
Dependency:
  xxhsum, md5sum, sha1sum, sha256sum, sha512sum and crc32
See also:
  man name_hash
  https://avpres.net/Bash_AVpres/
About:
  Abstract: Add, verify or remove a checksum as filename's suffix.
  Version:  2023-03-26

[10:25:36]imac01@iMac01:~/Desktop$
```

21

```
[10:23:55]imac01@iMac01:~/Desktop$ name_hash
Help:
  name_hash -h
[10:25:09]imac01@iMac01:~/Desktop$ name_hash -h

Usage:
  name_hash (-a|-v|-r) <input_path>
  name_hash -h | -x
Options:
  -a  add checksum to filename
  -v  verify if file's content and checksum matches
  -r  remove checksum from filename
  -h  this help
  -x  advanced options with their default arguments
Dependency:
  xxhsum, md5sum, sha1sum, sha256sum, sha512sum and crc32
See also:
  man name_hash
  https://avpres.net/Bash_AVpres/
About:
  Abstract: Add, verify or remove a checksum as filename's suffix.
  Version:  2023-03-26

[10:25:36]imac01@iMac01:~/Desktop$ man name_hash
```

22

```
name_hash(1)                Bash Scripts for AVpres                name_hash(1)

NAME
       name_hash - Add, verify or remove a checksum as a suffix of the
       filename

SYNOPSIS
       name_hash (-a|-v|-r) input_path

       name_hash -h | -x

DESCRIPTION
       Bash AVpres is a collection of Bash scripts for audio-visual
       preservation. One of these small programs is name_hash which can
       perform three different tasks:

       - add a checksum to the filename;
       - verify if the integrated checksum matches the file content;
       - remove the checksum from the filename.

       The default format is:
           path/to/filename_checksum.extension

       The script dismisses the necessity of handling additional manifest
:
```

23

```
OPTIONS
   BASIC OPTIONS
       -a input_path, --add=input_path
           path to a file or a folder

           Add the checksum as suffix of the filename.

       -v input_path, --verify=input_path
           path to a file or a folder

           Verify if the checksum matches the file's content.

           Note that the verification is faster when the --algorithm option is
           provided, especially when MD5 or CRC-32 is used.

       -r input_path, --remove=input_path
           path to a file or a folder

           Remove the filename's suffix containing the checksum.

   ADVANCED OPTIONS
       The arguments of the advanced options can be overwritten by the user.
       Please remember that any string containing spaces must be quoted, or
       its spaces must be escaped.
:
```

24

ADVANCED OPTIONS

The arguments of the advanced options can be overwritten by the user. Please remember that any string containing spaces must be quoted, or its spaces must be escaped.

--algorithm=(xxh32|xxh64|xxh128|md5|sha1|sha256|sha512|crc32)

We advise to use a faster non-cryptographic hash functions, because we consider that, for archival purposes, there is no necessity to apply a more complex unkeyed cryptographic hash function. The algorithm name can be passed in upper or lower case letters.

The default algorithm is xxHash 128:

--algorithm='xxh128'

Note that until end of 2020 the default algorithm was MD5, which has the same checksum size than the xxHash 128 algorithm. Therefore, if you verify files with an MD5 checksum, then you may pass the option **--algorithm=md5** in order to speed-up the verification.

Also xxHash 32 and CRC-32 have the same checksum size. If the algorithm is not specified, then xxHash 32 is checked before CRC-32.

:

SHA-512 command

--crc32='/bin/crc32'

CRC-32 command

--confirmation=(yes|no)

By default, the script demands confirmation from the user before removing any checksum. The option **--confirmation=no** avoids it, which is useful when the script is run in a batch process and/or chain of scripts.

--with_algorithm=(yes|no)

By default, the script does not include the algorithm in the filename. By setting the option **--with_algorithm=yes** the filename becomes:

path/to/filename_algorithm_checksum.extension

INFORMATIVE OPTIONS

-h, --help

display a help message

-x, --options

display the advanced options with their default arguments

:

NOTES

It is possible to have multiple checksum suffixes, separated by an underscore, but **name_hash** considers always the last one.

CONFIGURATION FILE

An external configuration file

${HOME}/.config/AVpres/Bash_AVpres/name_hash.txt

can be defined, allowing the script to import alternate default values for the following options:

default_algorithm
md5
sha1
sha256
sha512
xxh32
xxh64
xxh128
crc32
confirmation
with_algorithm

:

LOG FILES

Temporary log files are stored at

/tmp/AVpres/name_hash.XXXXXXXXXX

The log files can be used for debugging, for example by running **cat** on the address prompted with fatal error messages:

cat /tmp/AVpres/name_hash.XXXXXXXXXX

SEE ALSO

Yann Collet: "xxHash fast digest algorithm", version 0.1.1, 2018-10-10
https://github.com/Cyan4973/xxHash/blob/dev/doc/xxhash_spec.md

RFC 1321, "The MD5 Message-Digest Algorithm", April 1992
https://www.rfc-editor.org/info/rfc1321

RFC 3174, "US Secure Hash Algorithm (SHA1)", September 2001
https://www.rfc-editor.org/info/rfc3174

"Descriptions of SHA-256, SHA-384, and SHA-512"
https://web.archive.org/web/20130526224224/http://csrc.nist.gov/groups/STM/cavp/documents/shs/sha256-384-512.pdf

:

RFC 1321, "The MD5 Message-Digest Algorithm", April 1992
https://www.rfc-editor.org/info/rfc1321

RFC 3174, "US Secure Hash Algorithm (SHA1)", September 2001
https://www.rfc-editor.org/info/rfc3174

"Descriptions of SHA-256, SHA-384, and SHA-512"
https://web.archive.org/web/20130526224224/http://csrc.nist.gov/groups/STM/cavp/documents/shs/sha256-384-512.pdf

xxhsum(1), **md5sum**(1), **sha1sum**(1), **sha256sum**(1) and **sha512sum**(1).

COPYRIGHT

Copyright (c) 2014-2023 by Reto Kromer

LICENSE

The **name_hash** Bash script is released under a 3-Clause BSD License.

DISCLAIMER

The **name_hash** Bash script is provided "as is" without warranty or support of any kind.

2023-03-26 https://avpres.net/Bash_AVpres/ name_hash(1)
(END)

29

```
[10:35:33]imac01@iMac01:~/Desktop$ touch my_file.txt
[10:35:51]imac01@iMac01:~/Desktop$
```

30

```
[10:35:33]imac01@iMac01:~/Desktop$ touch my_file.txt
[10:35:51]imac01@iMac01:~/Desktop$ name_hash -a my_file.txt
my_file_99aa06d3014798d86001c324468d497f.txt
[10:36:25]imac01@iMac01:~/Desktop$
```

31

```
[10:35:33]imac01@iMac01:~/Desktop$ touch my_file.txt
[10:35:51]imac01@iMac01:~/Desktop$ name_hash -a my_file.txt
my_file_99aa06d3014798d86001c324468d497f.txt
[10:36:25]imac01@iMac01:~/Desktop$ name_hash -v my_file_99aa06d3014798d86001c324
468d497f.txt
my_file_99aa06d3014798d86001c324468d497f.txt
[10:36:57]imac01@iMac01:~/Desktop$
```

32

```
[10:35:33]imac01@iMac01:~/Desktop$ touch my_file.txt
[10:35:51]imac01@iMac01:~/Desktop$ name_hash -a my_file.txt
my_file_99aa06d3014798d86001c324468d497f.txt
[10:36:25]imac01@iMac01:~/Desktop$ name_hash -v my_file_99aa06d3014798d86001c324
468d497f.txt
my_file_99aa06d3014798d86001c324468d497f.txt
[10:36:57]imac01@iMac01:~/Desktop$ mv my_file_99aa06d3014798d86001c324468d497f.t
xt my_file_99aa06d3014798d86001c324468d497e.txt
[10:37:32]imac01@iMac01:~/Desktop$
```

33

```
[10:35:33]imac01@iMac01:~/Desktop$ touch my_file.txt
[10:35:51]imac01@iMac01:~/Desktop$ name_hash -a my_file.txt
my_file_99aa06d3014798d86001c324468d497f.txt
[10:36:25]imac01@iMac01:~/Desktop$ name_hash -v my_file_99aa06d3014798d86001c324
468d497f.txt
my_file_99aa06d3014798d86001c324468d497f.txt
[10:36:57]imac01@iMac01:~/Desktop$ mv my_file_99aa06d3014798d86001c324468d497f.t
xt my_file_99aa06d3014798d86001c324468d497e.txt
[10:37:32]imac01@iMac01:~/Desktop$ name_hash -v my_file_99aa06d3014798d86001c324
468d497e.txt
Error: 'my_file_99aa06d3014798d86001c324468d497e.txt' doesn't match.
[10:38:04]imac01@iMac01:~/Desktop$
```

34

```
[10:35:33]imac01@iMac01:~/Desktop$ touch my_file.txt
[10:35:51]imac01@iMac01:~/Desktop$ name_hash -a my_file.txt
my_file_99aa06d3014798d86001c324468d497f.txt
[10:36:25]imac01@iMac01:~/Desktop$ name_hash -v my_file_99aa06d3014798d86001c324
468d497f.txt
my_file_99aa06d3014798d86001c324468d497f.txt
[10:36:57]imac01@iMac01:~/Desktop$ mv my_file_99aa06d3014798d86001c324468d497f.t
xt my_file_99aa06d3014798d86001c324468d497e.txt
[10:37:32]imac01@iMac01:~/Desktop$ name_hash -v my_file_99aa06d3014798d86001c324
468d497e.txt
Error: 'my_file_99aa06d3014798d86001c324468d497e.txt' doesn't match.
[10:38:04]imac01@iMac01:~/Desktop$ name_hash -r my_file_99aa06d3014798d86001c324
468d497e.txt
Remove checksum from filename? (y|N) y
my_file.txt
[10:38:39]imac01@iMac01:~/Desktop$
```

35

```
[10:35:33]imac01@iMac01:~/Desktop$ touch my_file.txt
[10:35:51]imac01@iMac01:~/Desktop$ name_hash -a my_file.txt
my_file_99aa06d3014798d86001c324468d497f.txt
[10:36:25]imac01@iMac01:~/Desktop$ name_hash -v my_file_99aa06d3014798d86001c324
468d497f.txt
my_file_99aa06d3014798d86001c324468d497f.txt
[10:36:57]imac01@iMac01:~/Desktop$ mv my_file_99aa06d3014798d86001c324468d497f.t
xt my_file_99aa06d3014798d86001c324468d497e.txt
[10:37:32]imac01@iMac01:~/Desktop$ name_hash -v my_file_99aa06d3014798d86001c324
468d497e.txt
Error: 'my_file_99aa06d3014798d86001c324468d497e.txt' doesn't match.
[10:38:04]imac01@iMac01:~/Desktop$ name_hash -r my_file_99aa06d3014798d86001c324
468d497e.txt
Remove checksum from filename? (y|N) y
my_file.txt
[10:38:39]imac01@iMac01:~/Desktop$ ls -l /tmp/AVpres
total 48
-rw-------  1 imac01  wheel  285  4 Sep 10:36 name_hash.2xGXXWlgPP
-rw-------  1 imac01  wheel  200  4 Sep 10:25 name_hash.41KzKMkaaI
-rw-------  1 imac01  wheel  530  4 Sep 10:38 name_hash.GnyI9ElRml
-rw-------  1 imac01  wheel  337  4 Sep 10:36 name_hash.NnJhwcsJsr
-rw-------  1 imac01  wheel  288  4 Sep 10:38 name_hash.f48TCeUtPd
-rw-------  1 imac01  wheel  200  4 Sep 10:25 name_hash.vPmBnvbOgG
[10:39:11]imac01@iMac01:~/Desktop$
```

36

```
[10:40:40]imac01@iMac01:~/Desktop$ cat /tmp/AVpres/name_hash.GnyI9ElRml
[2023-09-04 08:38:04 UTC] name_hash 2023-03-26
[2023-09-04 08:38:04 UTC] /usr/local/bin/name_hash -v my_file_99aa06d3014798d860
01c324468d497e.txt
[2023-09-04 08:38:04 UTC] START
[2023-09-04 08:38:04 UTC] verify checksum
Trying 'xxh128'
1c1
< 99aa06d3014798d86001c324468d497e
---
> 99aa06d3014798d86001c324468d497f
Trying 'md5'
1c1
< 99aa06d3014798d86001c324468d497e
---
> d41d8cd98f00b204e9800998ecf8427e
[2023-09-04 08:38:04 UTC] Error: 'my_file_99aa06d3014798d86001c324468d497e.txt'
doesn't match.
[2023-09-04 08:38:04 UTC] END
[10:40:43]imac01@iMac01:~/Desktop$
```

37

```
[10:44:07]imac01@iMac01:~/Desktop$ name_hash -a my_file.txt --algorithm=md5
my_file_d41d8cd98f00b204e9800998ecf8427e.txt
[10:44:18]imac01@iMac01:~/Desktop$
```

38

```
[10:47:10]imac01@iMac01:~/Desktop$ name_hash -a my_file.txt --algorithm=md5
my_file_d41d8cd98f00b204e9800998ecf8427e.txt
[10:47:15]imac01@iMac01:~/Desktop$ name_hash -v /Users/imac01/Desktop/my_file_d4
1d8cd98f00b204e9800998ecf8427e.txt
my_file_d41d8cd98f00b204e9800998ecf8427e.txt
[10:47:26]imac01@iMac01:~/Desktop$ cat /tmp/AVpres/name_hash.92gLazc93H
[2023-09-04 08:47:26 UTC] name_hash 2023-03-26
[2023-09-04 08:47:26 UTC] /usr/local/bin/name_hash -v /Users/imac01/Desktop/my_f
ile_d41d8cd98f00b204e9800998ecf8427e.txt
[2023-09-04 08:47:26 UTC] START
[2023-09-04 08:47:26 UTC] verify checksum
Trying 'xxh128'
1c1
< d41d8cd98f00b204e9800998ecf8427e
---
> 99aa06d3014798d86001c324468d497f
Trying 'md5'
[2023-09-04 08:47:26 UTC] my_file_d41d8cd98f00b204e9800998ecf8427e.txt
[2023-09-04 08:47:26 UTC] END
[10:47:57]imac01@iMac01:~/Desktop$
```

39

```
[2023-09-04 08:47:26 UTC] name_hash 2023-03-26
[2023-09-04 08:47:26 UTC] /usr/local/bin/name_hash -v /Users/imac01/Desktop/my_f
ile_d41d8cd98f00b204e9800998ecf8427e.txt
[2023-09-04 08:47:26 UTC] START
[2023-09-04 08:47:26 UTC] verify checksum
Trying 'xxh128'
1c1
< d41d8cd98f00b204e9800998ecf8427e
---
> 99aa06d3014798d86001c324468d497f
Trying 'md5'
[2023-09-04 08:47:26 UTC] my_file_d41d8cd98f00b204e9800998ecf8427e.txt
[2023-09-04 08:47:26 UTC] END
[10:47:57]imac01@iMac01:~/Desktop$ name_hash -v /Users/imac01/Desktop/my_file_d4
1d8cd98f00b204e9800998ecf8427e.txt --algorithm=md5
my_file_d41d8cd98f00b204e9800998ecf8427e.txt
[10:48:36]imac01@iMac01:~/Desktop$ cat /tmp/AVpres/name_hash.n8dMa3EFMA
[2023-09-04 08:48:36 UTC] name_hash 2023-03-26
[2023-09-04 08:48:36 UTC] /usr/local/bin/name_hash -v /Users/imac01/Desktop/my_f
ile_d41d8cd98f00b204e9800998ecf8427e.txt --algorithm=md5
[2023-09-04 08:48:36 UTC] START
[2023-09-04 08:48:36 UTC] verify checksum
[2023-09-04 08:48:36 UTC] my_file_d41d8cd98f00b204e9800998ecf8427e.txt
[2023-09-04 08:48:36 UTC] END
[10:49:03]imac01@iMac01:~/Desktop$
```

40

File Generation

- FFV1
- ProRes
- Avid
- H.264 (AVC)
- H.265 (HEVC)
- H.266 (VVC)
- AV1

Missing Files


```
missing_files(1)          Bash Scripts for AVpres          missing_files(1)

NAME
       missing_files - Find missing files in a folder of sequentially numbered
       files

SYNOPSIS
       missing_files -i input_folder [-o output_file]

       missing_files -h | -x

DESCRIPTION
       Bash AVpres is a collection of Bash scripts for audio-visual
       preservation. One of these small programs is missing_files, which finds
       the missing files in a folder of sequentially numbered files.

       Bash version 3.2 or later is strongly recommended. We advise to use the
       current version 5.2.

OPTIONS
   BASIC OPTIONS
       -i input_folder, --input=input_folder
           input folder to analyse

:
```



```
USAGE
       The possibly generated file can be used, for example, to copy the
       missing frames from the source folder to the analysed input_folder in
       an automated way:

           for f in $(cat <input_folder>_missing.txt); do
             cp -v "<source_folder>/${f}" "<input_folder>"
           done

NOTES
       Note that the script verifies the presence of all the sequentially
       numbered filenames in an interval, but not the content of those files
       (for this checksums are needed, and their use is highly recommended).
       For obvious reasons, missing frames at the very beginning or end of the
       interval cannot be detected.

   CONFIGURATION FILE
       An external configuration file

           ${HOME}/.config/AVpres/Bash_AVpres/missing_files.txt

       can be defined, allowing the script to import alternate default value
       for the following option:
:
```

Metadata Extraction


```
metadata_csv(1)            Bash Scripts for AVpres            metadata_csv(1)

NAME
       metadata_csv - Recursively create a CSV file with metadata of all AV
       files in a folder and its subfolders

SYNOPSIS
       metadata_csv -i input_path [-o output_file]

       metadata_csv -h | -x

DESCRIPTION
       Bash AVpres is a collection of Bash scripts for audio-visual
       preservation. One of these small programs is metadata_csv. It creates
       recursively a CSV file with metadata of all AV files in a folder and
       its subfolders. The main container, video and audio codec data are
       extracted.

       Bash version 4.3 or later is recommended. We advise to use the current
       version 5.2.

OPTIONS
   BASIC OPTIONS
       -i input_path, --input=input_path
:
```



```
ADVANCED OPTIONS
    The arguments of the advanced options can be overwritten by the user.
    Please remember that any string containing spaces must be quoted, or
    its spaces must be escaped.

    --verbosity=(path|filename|count|off)
        For each processed file the Terminal will show:

            path     = full path
            filename = only filename and extension
            count    = a counter (default)
            off      = nothing, i.e. work silently

    --exclusion='^(extension_list)$'
        The extension_list is defined as a regex of the file extensions
        which are excluded from the analysis. The default value is:

            exclusion='^(txt|md5|xml|pdf|xlsx|xls|docx|doc|zip|gz)$'

        Note that only the actual extension is listed, without the
        separation period.

    --ffprobe='/bin/ffprobe'
:
```


BagIt

- make
- verify
- update
- modify
- undo

```
make_bagit(1)                Bash Scripts for AVpres                make_bagit(1)

NAME
       make_bagit - Create a BagIt archive of a folder, according to RFC 8493

SYNOPSIS
       make_bagit -b input_folder | -i input_folder -o output_folder

       make_bagit -h | -x

DESCRIPTION
       Bash AVpres is a collection of Bash scripts for audio-visual
       preservation. One of these small programs is make_bagit. It creates an
       archive of a folder, according to the "BagIt File Packaging Format", as
       designed by the Library of Congress in the U.S.A. and standardised
       under RFC 8493.

       Bash version 3.2 or later is strongly recommended. We advise to use the
       current version 5.2.

OPTIONS
   BASIC OPTIONS
       -b input_folder, --bagit=input_folder
           input folder
:
```

49

```
OPTIONS
   BASIC OPTIONS
       -b input_folder, --bagit=input_folder
           input folder

           This replaces the input_folder by its BagIt archive, containing the
           original folder, and is faster.

       -i input_folder, --input=input_folder
           input folder

       -o output_folder, --output=output_folder
           output BagIt folder

           This copies the content of the input_folder into the output_folder
           while creating a BagIt archive, and requires more time.

   ADVANCED OPTIONS
       The arguments of the advanced options can be overwritten by the user,
       either by passing a different value when calling the script, or by
       setting it in the configuration file. Please remember that any string
       containing spaces must be quoted or its spaces must be escaped.

       --bagit_version=bagit_version
:
```

50

```
           verification

       --algorithm=(md5|sha1|sha256|sha512|xxh128|xxh64|xxh32|crc32)
           RFC 8493 considers the four unkeyed cryptographic hash functions
           MD5, SHA-1, SHA-256 and SHA-512, which is the default:
               --algorithm=sha512

           make_bagit and verify_bagit support all four hash algorithms, but
           the use of different and/or multiple algorithms in the same BagIt
           archive is currently not supported.

           In addition, make_bagit and verify_bagit also support the four non-
           cryptographic algorithm xxh128, xxh64, xxh32 and crc32. Please note
           that this does not follow the RFC 8493 specification at all!

           The algorithm name can be passed in upper or lower case letters.

       --with_scripts=(yes|no)
           include or not the make_bagit, verify_bagit and undo_bagit scripts
           into the BagIt archive (default is no)

       The script needs an external command to compute recursively the hash
       checksums of all elements inside the folder.

:
```

51

FFmpeg Tools

- FFEngine presets
- FFmpeg head

52

Workflow ("Film")

DPX or TIFF source files

- generate archive file: Matroska/FFV1
- generate mezzanine file: QuickTime/ProRes
- generate access file: MP4/H.264
- generate checksum manifest
- generate BagIt

Other Approaches

- Microservices
- Miraculix

ARTICLE

Microservices in Audiovisual Archives: An Exploration of Constructing Microservices for Processing Archival Audiovisual Information

Annie Schweikert, New York University, USA
Dave Rice, City University of New York, USA
DOI: https://doi.org/10.35320/ij.v0i50.70

Abstract

Properly managing audiovisual archival material requires identifying, using, and possibly creating the right tools and workflows to facilitate archival objectives. In creating these workflows, two models are possible. One model is the monolithic architecture, which includes complex all-in-one systems (for instance, a comprehensive digital asset management system). Another model is the microservice architecture, which combines independent tools into a loosely coupled system based upon common underlying standards and understandings. In a microservice architecture, an individual tool may be added, replaced, or upgraded independently of the other tools.

This document describes and examines strategies for designing lightweight microservice environments for the processing of digital, file-based, audiovisual data within an archive. It guides the reader through the design of a simple example microservice architecture by establishing foundational archival frameworks for microservice design, describing examples of packages and microservices tailored to audiovisual archives, and finally demonstrating an end-to-end workflow.

This document does not intend to be a standard for the design of audiovisual microservices, but seeks to contribute a use case to the work and dialogue of many audio-visual archives exploring and implementing microservice structures; see in particularly

AV Preservation by
reto.ch

Sandrainstrasse 3
3007 Bern
Switzerland

reto.ch
info@reto.ch