

ObjectArchive

Pierre Carrette, Sylvain Rougier, Jean-Baptiste Yunes, and Bertrant LeCun

COLLABORATORS

	TITLE : ObjectArchive		
ACTION	NAME	DATE	SIGNATURE
WRITTEN BY	Pierre Carrette, Sylvain Rougier, Jean-Baptiste Yunes, and Bertrant LeCun	March 28, 2025	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	ObjectArchive	1
1.1	ObjectArchive V1.4 Documentation	1
1.2	ObjectArchive.guide/System requirements	2
1.3	ObjectArchive.guide/File Types and whatis.library	2
1.4	ObjectArchive.guide/Install	2
1.5	ObjectArchive.guide/Introduction	2
1.6	ObjectArchive.guide/History	3
1.7	ObjectArchive.guide/Using ObjectArchive	4
1.8	ObjectArchive.guide/Buffering	5
1.9	ObjectArchive.guide/Tar	6
1.10	ObjectArchive.guide/Compressed Tar	7
1.11	ObjectArchive.guide/Lha	7
1.12	ObjectArchive.guide/Developer Informations	8
1.13	ObjectArchive.guide/Known Problems	9
1.14	ObjectArchive.guide/Status	9
1.15	ObjectArchive.guide/Release Notes	10
1.16	ObjectArchive.guide/Updates	10
1.17	ObjectArchive.guide/Bug reports	11
1.18	ObjectArchive.guide/Disclaimer	11

Chapter 1

ObjectArchive

1.1 ObjectArchive V1.4 Documentation

Welcome to	
ObjectArchive	
A new standard for archive manipulation	
Version 1.4	
Copyright © 1994-95	
Pierre Carrette, Sylvain Rougier, Jean-Baptiste Yunes, Bertrant LeCun	
System requirements	
Installation	
Introduction	Start reading this.
The history	The project history since the first idea
Using ObjectArchive	Everyday usage
Buffering	Why buffering and how ?
Tar	Tar archive format
Compressed Tar	Compressed Tar archive format
Lha	Lha/Lharc archive format
Developer Info	Developping new objects
Status	Public domain & Copyright
Known problems	Bugs/conflits with other software.
Release Notes	Features history since v1.0.
Getting Updates	Where and how to get new versions.
Bug reports	Found some bugs ?
Disclaimer	Be warned!

1.2 ObjectArchive.guide/System requirements

System requirements

=====

ObjectArchive requires version 2.0 of the Amiga operating system or higher, and the file type recognition library `whatis.library`.

1.3 ObjectArchive.guide/File Types and `whatis.library`

File Types and `whatis.library`

=====

This library is responsible of file type identification in `archive.library`. It now quite know as some other PD software uses it.

The `whatis.library` comes as a separate package, even though the library itself and a simple configuration file comes with it. The complete distribution can be downloaded from [aminet](http://aminet.net), and makes part of our install disk.

Read `whatis.library` documentation for more information about configuration.

1.4 ObjectArchive.guide/Install

Installation

=====

ObjectArchive is distributed together with an Installer script, therefore making installation a breeze!

Just double-click on the 'Install-ObjectArchive' icon and the installation procedure is on it's way.

As ObjectArchive requires `whatis.library`, the installation script will install it if necessary. But the actual `whatis` configuration does not make it easy to install new file types. Then you may have to edit your `S:FileTypes` file manually.

ObjectArchive comes with three archive objects, Tar, TarGZ, and Lha. The Tar and TarGZ file types may not be part of your `whatis` configuration. You'll then have to add them.

The `archive.library` has an environnement variable "`ENV:Archives/Archive`". It is used for Buffering parameters.

1.5 ObjectArchive.guide/Introduction

Introduction

=====

ObjectArchive is an attempt to standardise archive usage. The most important feature is to give access to archives using a handler, making it possible to list, add, extract archive (just like if it was a disk), under any directory utility, workbench, or shell. After that, the extension or the idea was to be able to perform usual archive operations with the same syntax whichever the archive format is. It is very annoying having three or four common archivers with each their own command line syntax. Now, you may:

```
ArcExtract foobar #? ALL
```

and all the "foobar" archive contents would be extracted.

To access an archive through a handler, type the following command:

```
MountArchive foobar
```

If the format of the archive is recognised, a new volume foobar: will appear in the system, as well as a disk icon in the workbench. Read Using ObjectArchive for a complete usage description.

1.6 ObjectArchive.guide/History

History

=====

By the end of 1993 appeared a file manager called MTool, which had a new feature. It was able to display the internal contents of an archive as if it were just a simple directory.

Some people then asked for that feature in BrowserII, our concurrent file manager ;-)

I started thinking about that, but I was not convinced that doing such would satisfy myself, for two reasons. First, it would insert loads of crappy code in BrowserII. Second, user would then have to know that such a displayed directory would not be a real directory. It would not have been possible to 'CD' in this dir, nor running software really from here, because started programs would not find their files in the current dir,...

So, after some hours of brain storm, the idea was born.

What is an archive ? A root directory, with sub directories, and files.

So, THE solution was a handler.

I started speaking about that on the french Amiga mailing list, exposing the idea, and asked for some people who would be interested by helping me in developping the project. I was busy with BrowserII and had not much time.

Then, Jean-Baptiste Yunes and Bertrand LeCun joined our team (Sylvain and I ;-)

Of course, the first need was an lha handler. So, the wonderfull Jibounet (Jean Baptiste) created the lha mailing list, which have 4 subscribers!

The idea made it's course since then, and here's the result.

1.7 ObjectArchive.guide/Using ObjectArchive

Using ObjectArchive

=====

Using ObjectArchive is very simple.

Workbench

~~~~~

Set your archive icons default tool to "MountArchive".  
Then double-click on the archive, and a disk icon appears beside your usual harddisk partitions. Double-click on the archive disk, a window appear, with the archive contents! All normal file/drawer operations are available, file copying, tools running,... You may even mount archives from within other archives!!

#### Shell

~~~~~

Imagine you want to see the contents of an archive with your preferred directory listing command, let say "ls".
You have an archive foobar.lha in Work:PublicDomain.

```
1.Work:PublicDomain> MountArchive foobar.lha
1.Work:PublicDomain> CD foobar:
1.foobar:> ls -lR
```

That's it!

Now you have done with foobar.lha, you want to get rid of the foobar: volume. just:

```
1.Work:PublicDomain> UnMount foobar:
```

The volume will be unmounted as soon as there's no more locks/open files for that volume. It may immediately be unmounted!

Using in scripts

~~~~~

It may sometimes be usefull to create some script that would mount an archive in order to access it using the relative created volume.  
But getting the volume or device name from the archive name may not be easy, or even impossible. So, there is a SHOWDEV switch to the MountArchive command.

#### Example:

~~~~~

```
1.Work:PublicDomain> MountArchive foobar.lha SHOWDEV
LHA0:
1.Work:PublicDomain>
```

You can see the dos device name in which the volume foobar: is inserted.

Then, imagine your want to create a script to 'CD' in an archive.
You would:

```
CD 'MountArchive foobar.lha SHOWDEV' ; this would "CD LHA0:"
```

If you want the volume to be unmounted as soon as possible:

```
Set ArcDev 'MountArchive foobar.lha SHOWDEV'
CD $ArcDev
UnMount $ArcDev ; As we are in the volume, it won't be unmounted now.
UnSet $ArcDev
```

BrowserII

~~~~~

Read BrowserII documentation for that.

BrowserII has a direct archive.library support, so no special configuration is required. Archives are mounted and unmounted automatically when appropriate.

Note:

~~~~~

For now, archive disks are write-protected.

1.8 ObjectArchive.guide/Buffering

Buffering

=====

For maximum efficiency, the archive.library provides a Buffering mechanism global for all archives.

Why ?

~~~~~

Imagine you double-click on a file in BrowserII. The file is first open by whatis.library to determine it's type. Then, if BrowserII has an autocommand for this file, let say 'ViewTek' if file is an image, BrowserII runs ViewTek, which opens the file a second time and display it.

Without Buffering, the file would be unpacked twice. For simple archive objects which use an external archiver, such as the actual lha.archive, this would be definitely awful!

How ?

~~~~~

The Buffering can be memory or disk based. In both case, you set a maximum buffer space, depending on your system available resources. Then, buffers are allocated by archive objects when needed. They are automatically flushed when archives are unmounted. When buffer space is required, but maximum buffer space amount has been reached, unused buffers are flushed using an "LRU" algorithm, ie, the Least Recently Used first. Anyway, it may happen that too many buffers are in use, making it impossible to allocate new buffers without going off the limit. So, an attempt to allocate more buffers will be done. So, be prepared to that. This means that the partition to which disk buffers are written should have more space available than maximum disk buffer space you set.

Then, you have to ajust buffer parameters to your system.

1. The tmp directory for disk based buffers. (should be on a hard disk partition although default is T:).
2. The maximum disk space disk based buffers may use. The bigger the better.
3. The maximum memory space for memory based buffers.
(A future version will allow explicit virtual memory support.)

All these options can be modified using archive.library's environnement variable "ENV:Archives/Archive", which template is:

TmpDir/K,MaxTmpDiskSpace/K/N,MaxTmpMemorySpace/K/N

MaxTmpDiskSpace and MaxTmpMemorySpace are in KBytes.

Default is: "T:" 256 256

Example:

~~~~~

TmpDir=Work:Tmp MaxTmpDiskSpace=1024 MaxTmpMemorySpace=512

## 1.9 ObjectArchive.guide/Tar

Tar

===

Tar is an old format used for tape backups on unix systems. ObjectArchive comes with a Tar object mainly because it was very easy to implement as there is no data compression. It was a good base to develop and test the ObjectArchive architecture.

The Tar object will be copied into your system by the installation procedure. But, to be fonctionnal, the whatis.library must also be configured.

If you are new to this library, the installation then copied the appropriate configuration file "FileTypes" in your S: directory. If you already had such a file, the "Tar" file format may not be known. The actual whatis.library database does not allow easy installation of new file tyes. This will change in the next release. For now, you must check for the existence of the following file type description in your S:FileTypes file:

```
TYPE "Tar"
  OPTNAMEPATTERN "#?.tar"
  COMPAREBYTE 257 "ustar"
ENDTYPE
```

Note:

~~~~~

The Amiga port of the tar archiver GNUTar does not set the magic value "ustar" in file headers. If you are using GNUTar to create tar archive on your Amiga, your must then use this alternative description:

```

TYPE "Tar"
    NAMEPATTERN "#?.tar"
ENDTYPE

```

1.10 ObjectArchive.guide/Compressed Tar

Compressed Tar
=====

Compressed Tar object is derived from tar. It then needs it to work. In fact, it just decompresses the archive to a temporary tar file. The GZip command is required for that. As GZip decompresses GZip'ed as well as Compress'ed tar archives, the targz object also reads both.

The file type description for the S:FileTypes file is:

```

TYPE TarGZ
    NAMEPATTERN "#?.(Z|gz)"
ENDTYPE

```

In case the GZip command wouldn't be in your C: directory, you then need to create an environnement variable ENV:Archives/TarGZ. This is a text format variable, which template is:

GZIPPATH/K,TMPDIR/K

Example:

~~~~~

GZIPPATH=SYS:Utilities/GZip TMPDIR=Work:Tmp

The default TMPDIR is T:. But if you don't have much memory, as T: is by default located in RAM Disk, you may set your TMPDIR to a harddisk drawer, to be able to read big archives.

## 1.11 ObjectArchive.guide/Lha

Lha  
===

Who does not know lha ?

This version is temporary. It requires the external lha command for file extraction, except for "stored" files (not compressed). Lha does not compress already compressed files.

But archive directory reading is builtin. It is a lot faster than calling lha and decode output!!

In order to handle self extract lha archives, you must add the following file type to your S:FileTypes file:

```

TYPE "Lha SFX"

```

---

```

SUBTYPE "Exe"
INSERTAFTER "LhArc"
OPTNAMEPATTERN "#?.run"
COMPAREBYTE 44 "SFX!"
ENDTYPE

```

A version with builtin file extraction is under developpement.

Lha archive has an environnement variable ENV:Archives/Lha.  
 It is required if your lha command is somewhere else than in C:.  
 This is a text format variable, which template is:

LHAPATH/K

Example:

~~~~~

LHAPATH=SYS:Utilities/Lha

1.12 ObjectArchive.guide/Developer Informations

Developer Informations

=====

Software Organisation

~~~~~

ObjectArchive is composed of the following elements.

- A library:                archive.library        to put in your LIBS: directory.
- Shell commands        MountArchive        to put in your C: directory.  
                       UnMount  
                       ArcList                (not yet!)  
                       ArcAdd                (not yet!)  
                       ArcExtract            (not yet!)
- Archive objects        lha.archive        to put in SYS:Classes/Archives  
                       tar.archive  
                       targz.archive  
                       ...
- Archive descriptors    Lha                to put in DEVS:Archives  
                       Tar  
                       TarGZip  
                       ...

Your may have noticed that it is organised like the 3.x AmigaOS datatypes library. To the contrary of this last one, ObjectArchive is fonctionnal under AmigaOS 2.x.

#### Writing new objects

~~~~~

The goal of ObjectArchive was to make it easy writing archive objects.
 An object writer does not have to care about the handler, which would

need some dos documentation.

The developer has 6 mains methods to write. Three to examine the archive directory tree, and three more to read files. They are:

ReadFirstHeader	Start scanning archive, and give back first header
ReadNextHeader	Next header...
ReadHeaderEnd	Cleanup.
Open	Open a file (just some inits).
Read	Fill a buffer of bytes.
Close	Perform some cleanup.

Easy isn't it ? All standard dos functions are callable from theses methods so there should be any difficulty, except for (de)compression related topics.

Developer include files & documentation

~~~~~

ObjectArchive developer files are not yet available.

They will be available in a few days.

I still have the developer docs to write, and make sure the interface between the archive.library and the objects will allow to implement writing to archive with keeping compatibility with actual objects.

## 1.13 ObjectArchive.guide/Known Problems

Known Problems

=====

The lha object does not work if xData is running. A deadlock will occur on file extraction. This seems to be a limitation of xData, which prevents several processes to Open() simultaneously. No workaround is possible to get rid of all deadlock situations. Thanks to David Larson (KingCON author) for this explanation. If I can get xData sources files, I'll try to fix it...

To solve this, you may switch to another shareware "PDPro" which does the same as xData (and even more) and that seems to work properly together with ObjectArchive.

## 1.14 ObjectArchive.guide/Status

Status

=====

ObjectArchive is Freely Distributable.

It is copyrighted by Sylvain Rougier, Pierre Carrette, Jean-Baptiste Yunes, and Bertrand LeCun.

It may not be sold in or together with commercial software without written approval by the authors. It may be freely distributed on usual Fish disks

and cdroms, Aminet servers and CDROM, Pearl, CAM,...

It cannot be made available for downloading from a server which is a commercial operation. The general principle is that it is forbidden to make a profit through the use or distribution of this program without our written consent.

Distribution by "France Festival Distribution" after any sort of translation is definitely ILLEGAL.

Send donations to:

~~~~~

Sylvain Rougier
39 rue Carnot
86000 Poitiers
France.

1.15 ObjectArchive.guide/Release Notes

Release Notes

=====

- 1.0: First Release (22-Dec-94)
- 1.1: Buffering and Lha object.
- 1.2: Bug fixes. (Path to MountArchive was limited to 40 bytes)
Handler Requesters appear in calling application's screen.
Archives volume now have a dostype:
'LHA\0' for lha, 'TAR\0' for tar, and 'TARZ' for targz.
- 1.3: Added self extract lha archive support.
Bug fix in lha level 2 headers decode.
- 1.4: Replaced OpenFromLock() with Open() in all objects for handlers
that do not support new v37 packets (caused empty archive dirs).
Getting parent of an archive root directory now fails.
(ex: CD foo:/bar fails. CD foo:bar is correct)
No more need to terminate TMPDIR with a '/' for environnement
variable ENV:Archives/Archive.

1.16 ObjectArchive.guide/Updates

Updates

=====

To get the lastest version of ObjectArchive, download it from an aminet site in directory /pub/aminet/util/arc.

If you do not have an aminet access, you'll then have to wait for Fred Fish disks/cdroms, or other diskette collections.

1.17 ObjectArchive.guide/Bug reports

Bug reports

=====

If you have an internet access, the best way to send bug reports and enhancement requests is to send an e-mail to:

bvme@chasseneuil.em.slb.com (Pierre Carrette)

If you encounter a VALID archive (readable with the original archiver such as lha, tar,...) which has problems with ObjectArchive, consider sending it to me if it is not too large, please.

1.18 ObjectArchive.guide/Disclaimer

Disclaimer

=====

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDER AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.