wapmorgan / UnifiedArchive

UnifiedArchive - unified interface to archive (zip, 7z, rar, gz, bz2, xz, cab, tar, tar.gz, tar.bz2, tar.xz, tar.Z, iso) for listing, reading, extracting and creation + built-in console packer and unpacker + fully implemented PclZip-like interface (create, listContent, extract, properties, add, delete, merge, duplicate). http://wapmorgan.github.io/UnifiedArc...

#tar #zip #rar #archives #archiving #cab #iso #7zip #gzip #bzip2 #lzma2

87 commits | 1 branch | 10 releases | 2 contributors | MIT

Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

wapmorgan Update README.md — Latest commit `a51121f` on Dec 16, 2017

bin	Delete docopt from requirements	6 months ago
examples	Update example	a year ago
src	Add CAM as a part of UnifiedArchive	7 months ago
.gitignore	ADD LZW-compression support.	4 years ago
LICENSE	Update LICENSE	4 years ago
README.md	Update README.md	2 months ago
_config.yml	Set theme jekyll-theme-dinky	a year ago
composer.json	Fix composer	6 months ago
extendedSyntax.md	update readme	7 months ago

README.md

UnifiedArchive - unified interface to any archive (zip # 7z # rar # gz # bz2 # xz # cab # tar # tar.gz # tar.bz2 # tar.x # tar.Z # iso-9660) for listing, reading, extracting and creation + built-in console archive manager + fully implemented PclZip-like interface (create, listContent, extract, properties, add, delete, merge, duplicate).

packagist wapmorgan/unified-archive | stable 0.0.10 | downloads 36.77 k | license MIT

Contents:

Preamble

If on your site there is a possibility of uploading of archives and you would like to add functionality of their automatic unpacking and viewing with no dependency on format of the archive, you can use this library.

Installation

Composer package: `wapmorgan/unified-archive` [1]

```
{
    "require": {
        "wapmorgan/unified-archive": "~0.0.10"
    }
}
```

Process of archive reading

1. Import a class

   ```
   require 'vendor/autoload.php';
   use \wapmorgan\UnifiedArchive\UnifiedArchive;
   ```

2. At the beginning, try to open the file with automatic detection of a format by name. In case of successful recognition a `UnifiedArchive` object will be returned. In case of failure - **null**

   ```
   $archive = UnifiedArchive::open('filename.rar');
   // or
   $archive = UnifiedArchive::open('filename.zip');
   // or
   $archive = UnifiedArchive::open('filename.7z');
   // or
   $archive = UnifiedArchive::open('filename.gz');
   // or
   $archive = UnifiedArchive::open('filename.bz2');
   // or
   $archive = UnifiedArchive::open('filename.xz');
   // or
   $archive = UnifiedArchive::open('filename.cab');
   // or
   $archive = UnifiedArchive::open('filename.tar');
   // or
   $archive = UnifiedArchive::open('filename.tar.gz');
   // or
   $archive = UnifiedArchive::open('filename.tar.bz2');
   // or
   $archive = UnifiedArchive::open('filename.tar.xz');
   // or
   $archive = UnifiedArchive::open('filename.tar.Z');
   // or
   $archive = UnifiedArchive::open('filename.iso');
   ```

3. Further, read the list of files of archive (note: this function returns only names of files)

   ```
   var_dump($archive->getFileNames());
   ```

4. Further, you can get additional information about concrete file by `getFileData()` method

   ```
   var_dump($archive->getFileData('README.md'));
   ```

5. Further, you can get raw file contents by `getFileContent()` method

   ```
   var_dump($archive->getFileContent('README.md'));
   ```

6. Further, you can unpack any internal catalog or the whole archive with files on a disk. The `extractNode()` method is engaged in it. In case of success, it returns number of the extracted files, in case of failure - **false**. Initial and final symbol of division of catalogs are very important! Don't forget them.

   ```
   $archive->extractNode(outputFolder, archiveFolder = '/');
   // to unpack all contents of archive
   ```

```
$archive->extractNode(tmpnam('/tmp', 'arc'));
// to unpack the src catalog in archive in the sources catalog on a disk
$archive->extractNode(tmpnam('/tmp', 'sources'), '/src/');
// to unpack the bookmarks catalog in archive in the sources catalog on a
// disk
$archive->extractNode(tmpnam('/tmp', 'sources'), '/bookmarks/');
```

Process of archive modification

To delete a single file from an archive:

```
$archive->deleteFiles('README.md');
```

To delete multiple files from an archive

```
$archive->deleteFiles(array('README.md', 'MANIFEST.MF'));
```

In case of success the number of successfully deleted files will be returned.

Process of archive addition

To add completely the catalog with all attached files and subdirectories:

```
$archive->addFiles('/var/log');
```

To add one file:

```
$archive->addFiles('/var/log/syslog');
```

To add some files or catalogs:

```
$archive->addFiles(array(directory, file, file2, ...));
```

Full syntax of multiple files addition is described in another document.

Process of archive creation & modification

To pack completely the catalog with all attached files and subdirectories in new archive:

```
UnifiedArchive::archiveNodes('/var/log', 'Archive.zip');
```

To pack one file:

```
UnifiedArchive::archiveNodes('/var/log/syslog', 'Archive.zip');
```

To pack some files or catalogs:

```
UnifiedArchive::archiveNodes(array(directory, file, file2, ...), 'Archive.zip');
```

Extended syntax with possibility of rewriting of paths and additional opportunities:

```
$nodes = array(
    array('source' => '/etc/php5/fpm/php.ini', 'destination' => 'php.ini'),
    array('source' => '/home/.../Dropbox/software/1/',
        'destination' => 'SoftwareVersions/', 'recursive' => true),
    array('source' => '/home/.../Dropbox/software/2/',
        'destination' => 'SoftwareVersions/', 'recursive' => true),
    array('source' => 'pictures/other/cats/*', 'destination' => 'Pictures/'),
    array('source' => '~/Desktop/catties/*', 'destination' => 'Pictures/'),
    array('source' => '/media/wapmorgan/.../Cats/*',
        'destination' => 'Pictures/'),
);
UnifiedArchive::archiveNodes($nodes, 'Archive.zip');
```

Complete description of extended syntax with examples and explanations..

Restrictions

It is impossible to create the ideal archiver, here therefore some restrictions and technical features are listed further:

1. Creation of *rar*, *iso* archives is impossible due to format limits.
2. If the `source` doesn't end with `/` or `*`, it means to be a file.
3. If the `source` is a directory, destination too means has to be a directory (to terminate on "/").

Built-in archiver

To see all opportunities of this remarkable UnifiedArchive, together with source codes some scripts are delivered. They can become quite good replacement to standard commands of file system (tar, unzip, rar, gzip).

Format	Program usage	Replacement
tar	`tar xv -C $DIRECTORY archive.tar.gz`	`./cli.hierarchy.php -e -n / -a archive.tar.gz -o $DIRECTORY`
zip	`unzip archive.zip -d $DIRECTORY`	`./cli.hierarchy.php -e -n / -a archive.zip -o $DIRECTORY`
rar	`unrar x archive.rar $DIRECTORY`	`./cli.hierarchy.php -e -n / -a archive.rar -o $DIRECTORY`
gzip	`gzip -d -k archive.gz && mv archive $DIRECTORY`	`./cli.hierarchy.php -e -n / -a archive.gz -o $DIRECTORY`

You noticed? The universal extractor itself defines type of archive and there is no need manually to choose type.

We will continue and look at examples of replacement of the unbridled number of utilities on one command:

Operation	Command	Notes
List nodes in archive	`./cli.hierarchy.php -l -a archivename`	
Table nodes in archive	`./cli.hierarchy.php -t -a archivename`	
Hierarchy of nodes in archive	`./cli.hierarchy.php -h -a archivename`	
Details about a file	`./cli.hierarchy.php -d -a archivename -n file_in_archive`	Replaces `unzip -v archivename file_in_archive replacement`
Extract a file	`./cli.hierarchy.php -e -a archivename -n /file_in_archive -o .`	Replaces `unzip archivename file_in_archive`
Prints file content without extraction	`./cli.hierarchy.php -u -a archivename -n file_in_archive`	Replaces `unzip -p archivename file_in_archive`
Extract a folder	`./cli.hierarchy.php -e -a archivename -n /directory_in_archive/ -o .`	Replaces `unzip archivename "directory_in_archive/*"`
Archive information	`./cli.hierarchy.php -i -a archivename`	

In the future probably I will add still some scripts or I will update existing with new functions.

It is very easy and equally universal for any of supported types of archives (which already about 13)!

Try, it will be pleasant to you!

It is possible even to use the console version of the archiver in daily work for viewing of contents of archive from the terminal!

Conveniently, isn't that so?

Built-in console archive manager

UnifiedArchive is distributed with a unified console program to manipulate popular archive formats. This script is stored in `bin/cam`.

It supports all formats that UnifiedArchive does and can be used to manipulate archives without other software. To check your configuration and check formats support launch it with `-f` flag in console:

```
$ php bin/cam -f
```

Full usage help

```
USAGE: cam (-l|--list)  ARCHIVE
       cam (-t|--table) ARCHIVE
       cam (-i|--info)  ARCHIVE
       cam (-e|--extract) [--output=DIR] [--replace=(all|ask|none|time|size)] [--flat=(file|path)] [--
exclude=PATTERN] ARCHIVE [FILES_IN_ARCHIVE...]
       cam (-p|--print)   ARCHIVE FILES_IN_ARCHIVE...
       cam (-d|--details) ARCHIVE FILES_IN_ARCHIVE...
       cam (-x|--delete)  ARCHIVE FILES_IN_ARCHIVE...
       cam (-a|--add)     ARCHIVE FILES_ON_DISK...
       cam (-c|--create)  ARCHIVE FILES_ON_DISK...
       cam (-f|--formats)

ACTIONS:
      -l(--list)    List files
      -t(--table)   List files in table
      -i(--info)    Summary about archive

      -e(--extract) Extract from archive

      -p(--print)   Print files' content
      -d(--details) Details about files
      -x(--delete)  Delete files

      -a(--add)     Add to archive
      -c(--create)  Create new archive
```

API

Create object of class `\wapmorgan\UnifiedArchive\UnifiedArchive` implementing the `\wapmorgan\UnifiedArchive\AbstractArchive` interface which has the following methods:

Object methods

```
public function __construct($filename, $type)
```

Creation of object of a class.

```
public function getFileNames(): array
```

Obtaining the list of files in archive. The symbol of division of catalogs can be both a slash, and a backslash (depends on format).

```
public function getFileData($filename): stdClass
```

Obtaining detailed information on the file in archive. The name of the file has to COINCIDE in ACCURACY with one stored in archive. It is restricted to change a symbol of division of catalogs. This method returns object of **stdClass** with following fields:

- `string $filename` - file name in archive.
- `integer $compressed_size` - the size of the PACKED contents of the file.

- `integer $uncompressed_size` - the size of the UNPACKED contents of the file.
- `integer $mtime` - time of change of the file (the integer value containing number of seconds passed since the beginning of an era of Unix).
- `boolean $is_compressed` - the boolean value, containing **true** if the file was packed with compression.

```
public function getFileContent($filename): string
```

Receiving "crude" contents of the file. For text files it is the text, for images/video/music are crude binary data. The file name besides has to be in accuracy as in archive.

```
public function countFiles(): integer
```

Counts total of all files in archive.

```
public function getArchiveSize(): integer
```

Counts the archive size (the file size).

```
public function getArchiveType(): string
```

Receives archive type (like `rar` or `zip`).

```
public function countCompressedFilesSize(): integer
```

Counts the size of all PACKED useful data (that is contents of all files listed in archive).

```
public function countUncompressedFilesSize(): integer
```

Counts the size of all UNPACKED useful data (that is contents of all files listed in archive).

```
public function extractNode($outputFolder, $node = '/'): integer
```

Unpacks any of internal catalogs archive with full preservation of structure of catalogs in the catalog on a hard disk. Returns number of extracted files.

```
public function deleteFiles($fileOrFiles): integer
```

Updates existing archive by removing files from it. Returns number of deleted files.

```
public function addFiles($nodes): integer
```

Updates existing archive by adding new files. Returns total number of files after addition.

Static methods

```
static public function open($filename): UnifiedArchive | null
```

Tries to distinguish type of archive and returns a `UnifiedArchive` instance in case of success, **null** in case of failure.

```
static public function archiveNodes($nodes, $aname, $simulation = false): integer | boolean | array
```

Archives nodes transferred in the first argument. Returns number of the archived files in case of success, in case of failure - **false**. If as the third argument is **true**, then the real archiving doesn't happen, and the result contains the list of the files chosen for an archiving, their number and total size.

PclZip-like interface

UnifedArchive provides full realization of the interface known by popular archiving library "PclZip" (only for **zip** format, the last version 2.8.2).

Let's look at it:

```
use wapmorgan\UnifiedArchive\UnifiedArchive;
require 'vendor/autoload.php';
$archive = UnifiedArchive::open('ziparchive.zip');
$pclzip = $archive->pclzipInteface();
```

You are from this point free to use all available methods provided by the class PclZip:

1. `create()` - creation of new archive, packing of files and catalogs.
2. `listContent()` - receiving contents of archive.
3. `extract()` - unpacking of files and catalogs.
4. `properties()` - obtaining information on archive.
5. `add()` - addition of files in archive.
6. `delete()` - cleaning of archive of files.
7. `merge()` - "pasting" of two archives.
8. `duplicate()` - archive cloning.

All available options and the parameters accepted by original PclZip are also available.

It is also important to note increase in productivity when using my version of the PclZip-interface using a native class for work, over old and working with "crude" contents of archive means of the PHP-interpreter.

The PclZip-interface is at present in a stage of experimental realization. I ask to take it into account.

For those who isn't familiar with the PclZip interface or wishes to refresh knowledge, visit official documentation on PclZip on the official site: http://www.phpconcept.net/pclzip.

Also I need to note that one of an option nevertheless is unrealizable: *PCLZIP_OPT_NO_COMPRESSION*. This option allows to disconnect compression for added files. At present the native library for work *doesn't allow* to change compression parameters from zip-archive - all added the file forcibly contract. I tried to find a roundabout way, but at present to make it it didn't turn out.

Performance comparation

To confirm my words about boost that UnifiedArchive can make in your project, here's comparation table of UnifiedArchive and PclZip extracting the same archives.

Filename	UA (time)	% of PZ	PZ (time)
googletools.zip	0.014	**67%**	0.020
PHPWord-develop.zip	0.573	**63%**	0.907
turbosale_1.0.0.zip	0.250	**80%**	0.309
meteor-devel.zip	6.553	**62%**	10.429
subrion-develop.zip	10.682	**82%**	12.996
OptiKey-master.zip	3.445	**82%**	4.180

Average growth is 27%!

Examples

In the **examples** catalog there are some files for check of operability of the program. start them from a command line.

- Pass a catalog name to `cli.fs.php` script to scan directory for all archives.
- Pass a path to archive to `cli.hierachy.php` script for unpacking / listing.
- Pass a path to archive and file names to `cli.pack.php` script for archiving.

Progress of supporting various formats and operations

- One-file archives:
  - gzip - **creation** supported
  - bzip2 - **creation** supported
  - lzma2 - **creation** supported
- Proprietary archive formats:
  - rar - **only reading** from archive
- Other restrictions:
  - tar, tar.gz, tar.bz2, tar.xz, tar.Z - **creation and addition** supported, but not deletion (due to library restrictions).
  - iso - **only reading** from archive
  - cab - **only reading** from archive. **Extracting** supported only on speicific interpreter versions (>=7.0.22, >=7.1.8, >=7.2.0) due to bug in previous versions.
- General archive formats:
  - zip - **full support** (creation, addition, update)
  - 7zip - **full support** (creation, addition, update)

Changelog

Version	Date	Changelog
0.0.10	Aug 7, 2017	* Remove `docopt` from requirements.
0.0.9	Jul 20, 2017	* Added cam script.
0.0.8	Jan 24, 2017	* Added initial support for CAB archives without extracting. * Added handling of short names of tar archives. * Removed external repository declaration. * Removed die() in source code.
0.0.7	Jan 14, 2017	* Fixed using ereg function on PHP >7.
0.0.6	Jan 9, 2017	* Added functionality for adding files in archive. * Added functionality for deleting files from archive. * Fixed discovering 7z archive number of files and creating new archive.
0.0.5	Jan 8, 2017	* Added support for `7z` (7zip) archives.
0.0.4	Jan 7, 2017	* Added support for single-file `bz2` (bzip2) and `xz` (lzma2) archives.
0.0.3	Aug 18, 2015	* Removed archive_tar from required packages.
0.0.2	May 27, 2014	* Released under the MIT license
0.0.1	May 26, 2014	---