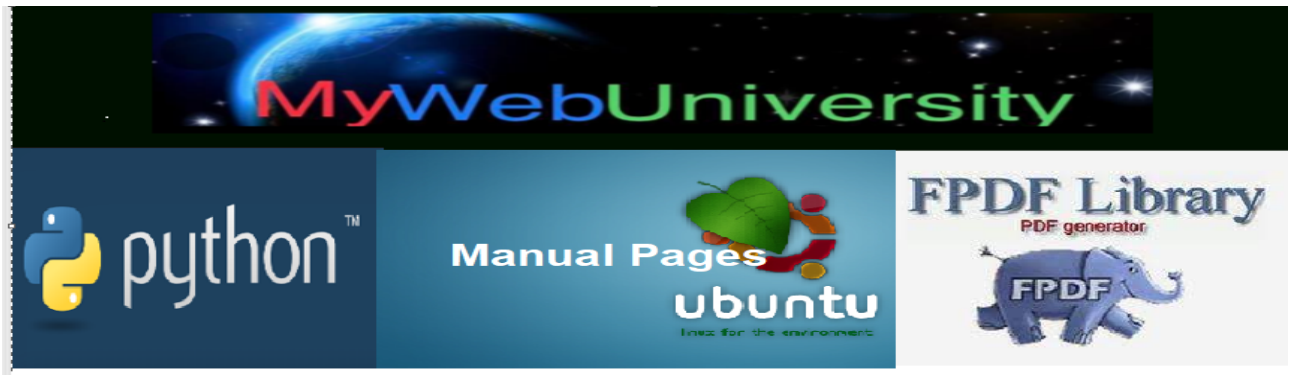

Full credit is given to the above companies including the OS that this PDF file was generated!

Linux Ubuntu 26.04 Manual Pages on command 'zipdetails.1'

$ man zipdetails.1

ZIPDETAILS(1) Perl Programmers Reference Guide ZIPDETAILS(1)

NAME

zipdetails - display the internal structure of zip files

SYNOPSIS

zipdetails [options] zipfile.zip

DESCRIPTION

This program creates a detailed report on the internal structure of zip files. For each item of metadata within a zip file the program will output

the offset into the zip file where the item is located.

a textual representation for the item.

an optional hex dump of the item.

The program assumes a prior understanding of the internal structure of Zip files. You should have a copy of the zip file definition, APPNOTE.TXT <https://pkware.cachefly.net/webdocs/casestudies/APPNOTE.TXT>, at hand to help understand the output from this program.

Default Behaviour

By default the program expects to be given a well-formed zip file. It will navigate the zip

file by first parsing the zip "Central Directory" at the end of the file. If the "Central Directory" is found, it will then walk sequentally through the zip records starting at the beginning of the file. See "Advanced Analysis" for other processing options.

If the program finds any structural or portability issues with the zip file it will print a message at the point it finds the issue and/or in a summary at the end of the output report. Whilst the set of issues that can be detected it exhaustive, don't assume that this program can find all the possible issues in a zip file - there are likely edge conditions that need to be addressed.

If you have suggestions for use-cases where this could be enhanced please consider creating an enhancement request (see "SUPPORT").

Date & Time fields

Date/time fields found in zip files are displayed in local time. Use the "--utc" option to display these fields in Coordinated Universal Time (UTC).

Filenames & Comments

Filenames and comments are decoded/encoded using the default system encoding of the host running "zipdetails". When the sytem encoding cannot be determined "cp437" will be used.

The exceptions are

? when the "Language Encoding Flag" is set in the zip file, the filename/comment fields are assumed to be encoded in UTF-8.

? the definition for the metadata field implies UTF-8 charset encoding

See "Filename Encoding Issues" and "Filename & Comment Encoding Options" for ways to control the encoding of filename/comment fields.

OPTIONS

General Options

"-h", "--help"

Display help

"--redact"

Obscure filenames and payload data in the output. Handy for the use case where the zip files contains sensitive data that cannot be shared.

"--scan"

Pessimistically scan the zip file loking for possible zip records. Can be error-prone. For very large zip files this option is slow. Consider using the "--walk" option first. See "Advanced Analysis Options"

"--utc"

By default, date/time fields are displayed in local time. Use this option to display them in in Coordinated Universal Time (UTC).

"-v" Enable Verbose mode. See "Verbose Output".

"--version"

Display version number of the program and exit.

"--walk"

Optimistically walk the zip file looking for possible zip records. See "Advanced Analysis Options"

Filename & Comment Encoding Options

See "Filename Encoding Issues"

"--encoding name"

Use encoding "name" when reading filenames/comments from the zip file.

When this option is not specified the default the system encoding is used.

" --no-encoding"

Disable all filename & comment encoding/decoding. Filenames/comments are processed as byte streams.

This option is not enabled by default.

"--output-encoding name"

Use encoding "name" when writing filename/comments to the display. By default the system encoding will be used.

"--language-encoding", "--no-language-encoding"

Modern zip files set a metadata entry in zip files, called the "Language encoding flag", when they write filenames/comments encoded in UTF-8.

Occasionally some applications set the "Language Encoding Flag" but write data that is not UTF-8 in the filename/comment fields of the zip file. This will usually result in garbled text being output for the filenames/comments.

To deal with this use-case, set the "--no-language-encoding" option and, if needed, set the "--encoding name" option to encoding actually used.

Default is "--language-encoding".

"--debug-encoding"

Display extra debugging info when a filename/comment encoding has changed.

Message Control Options

"--messages", "--no-messages"

Enable/disable the output of all info/warning/error messages.

Disabling messages means that no checks are carried out to check that the zip file is well-formed.

Default is enabled.

"--exit-bitmask", "--no-exit-bitmask"

Enable/disable exit status bitmask for messages. Default disabled. Bitmask values are: 1 for info, 2 for warning and 4 for error.

Default Output

By default "zipdetails" will output each metadata field from the zip file in three columns.

1. The offset, in hex, to the start of the field relative to the beginning of the file.

2. The name of the field.

3. Detailed information about the contents of the field. The format depends on the type of data:

   ? Numeric Values

   If the field contains an 8-bit, 16-bit, 32-bit or 64-bit numeric value, it will be displayed in both hex and decimal -- for example ""002A (42)"".

   Note that Zip files store most numeric values in little-endian encoding (there area few rare instances where big-endian is used). The value read from the zip file will have the endian encoding removed before being displayed.

   Next, is an optional description of what the numeric value means.

   ? String

If the field corresponds to a printable string, it will be output enclosed in single quotes.

? Binary Data

The term Binary Data is just a catch-all for all other metadata in the zip file. This data is displayed as a series of ascii-hex byte values in the same order they are stored in the zip file.

For example, assuming you have a zip file, "test,zip", with one entry

```
$ unzip -l  test.zip
Archive:  test.zip
Length      Date    Time    Name
---------  ---------- -----   ----
    446  2023-03-22 20:03   lorem.txt
---------                     -------
    446                     1 file
```

Running "zipdetails" will gives this output

```
$ zipdetails test.zip

0000 LOCAL HEADER #1       04034B50 (67324752)
0004 Extract Zip Spec      14 (20) '2.0'
0005 Extract OS            00 (0) 'MS-DOS'
0006 General Purpose Flag  0000 (0)
    [Bits 1-2]             0 'Normal Compression'
0008 Compression Method    0008 (8) 'Deflated'
000A Modification Time     5676A072 (1450614898) 'Wed Mar 22 20:03:36 2023'
000E CRC                   F90EE7FF (4178503679)
0012 Compressed Size       0000010E (270)
```

0016 Uncompressed Size 000001BE (446)

001A Filename Length 0009 (9)

001C Extra Length 0000 (0)

001E Filename 'lorem.txt'

0027 PAYLOAD

0135 CENTRAL HEADER #1 02014B50 (33639248)

0139 Created Zip Spec 1E (30) '3.0'

013A Created OS 03 (3) 'Unix'

013B Extract Zip Spec 14 (20) '2.0'

013C Extract OS 00 (0) 'MS-DOS'

013D General Purpose Flag 0000 (0)

[Bits 1-2] 0 'Normal Compression'

013F Compression Method 0008 (8) 'Deflated'

0141 Modification Time 5676A072 (1450614898) 'Wed Mar 22 20:03:36 2023'

0145 CRC F90EE7FF (4178503679)

0149 Compressed Size 0000010E (270)

014D Uncompressed Size 000001BE (446)

0151 Filename Length 0009 (9)

0153 Extra Length 0000 (0)

0155 Comment Length 0000 (0)

0157 Disk Start 0000 (0)

0159 Int File Attributes 0001 (1)

[Bit 0] 1 'Text Data'

015B Ext File Attributes 81ED0000 (2179792896)

[Bits 16-24] 01ED (493) 'Unix attrib: rwxr-xr-x'

[Bits 28-31] 08 (8) 'Regular File'

015F Local Header Offset 00000000 (0)

0163 Filename 'lorem.txt'

016C END CENTRAL HEADER 06054B50 (101010256)

0170 Number of this disk 0000 (0)

0172 Central Dir Disk no 0000 (0)

```
0174 Entries in this disk  0001 (1)
0176 Total Entries         0001 (1)
0178 Size of Central Dir   00000037 (55)
017C Offset to Central Dir 00000135 (309)
0180 Comment Length        0000 (0)
#
# Done
```

Verbose Output

If the "-v" option is present, the metadata output is split into the following columns:

1. The offset, in hex, to the start of the field relative to the beginning of the file.

2. The offset, in hex, to the end of the field relative to the beginning of the file.

3. The length, in hex, of the field.

4. A hex dump of the bytes in field in the order they are stored in the zip file.

5. A textual description of the field.

6. Information about the contents of the field. See the description in the "Default Output" for more details.

Here is the same zip file, "test.zip", dumped using the "zipdetails" "-v" option:

```
$ zipdetails -v test.zip

0000 0003 0004 50 4B 03 04 LOCAL HEADER #1       04034B50 (67324752)
0004 0004 0001 14          Extract Zip Spec      14 (20) '2.0'
0005 0005 0001 00          Extract OS            00 (0) 'MS-DOS'
0006 0007 0002 00 00       General Purpose Flag  0000 (0)
                           [Bits 1-2]            0 'Normal Compression'
```

0008 0009 0002 08 00 Compression Method 0008 (8) 'Deflated'

000A 000D 0004 72 A0 76 56 Modification Time 5676A072 (1450614898) 'Wed Mar 22 20:03:36 2023'

000E 0011 0004 FF E7 0E F9 CRC F90EE7FF (4178503679)

0012 0015 0004 0E 01 00 00 Compressed Size 0000010E (270)

0016 0019 0004 BE 01 00 00 Uncompressed Size 000001BE (446)

001A 001B 0002 09 00 Filename Length 0009 (9)

001C 001D 0002 00 00 Extra Length 0000 (0)

001E 0026 0009 6C 6F 72 65 Filename 'lorem.txt'

6D 2E 74 78

74

0027 0134 010E ... PAYLOAD

0135 0138 0004 50 4B 01 02 CENTRAL HEADER #1 02014B50 (33639248)

0139 0139 0001 1E Created Zip Spec 1E (30) '3.0'

013A 013A 0001 03 Created OS 03 (3) 'Unix'

013B 013B 0001 14 Extract Zip Spec 14 (20) '2.0'

013C 013C 0001 00 Extract OS 00 (0) 'MS-DOS'

013D 013E 0002 00 00 General Purpose Flag 0000 (0)

[Bits 1-2] 0 'Normal Compression'

013F 0140 0002 08 00 Compression Method 0008 (8) 'Deflated'

0141 0144 0004 72 A0 76 56 Modification Time 5676A072 (1450614898) 'Wed Mar 22 20:03:36 2023'

0145 0148 0004 FF E7 0E F9 CRC F90EE7FF (4178503679)

0149 014C 0004 0E 01 00 00 Compressed Size 0000010E (270)

014D 0150 0004 BE 01 00 00 Uncompressed Size 000001BE (446)

0151 0152 0002 09 00 Filename Length 0009 (9)

0153 0154 0002 00 00 Extra Length 0000 (0)

0155 0156 0002 00 00 Comment Length 0000 (0)

0157 0158 0002 00 00 Disk Start 0000 (0)

0159 015A 0002 01 00 Int File Attributes 0001 (1)

[Bit 0] 1 'Text Data'

015B 015E 0004 00 00 ED 81 Ext File Attributes 81ED0000 (2179792896)

[Bits 16-24] 01ED (493) 'Unix attrib: rwxr-xr-x'

[Bits 28-31] 08 (8) 'Regular File'

```
    015F 0162 0004 00 00 00 00 Local Header Offset   00000000 (0)
    0163 016B 0009 6C 6F 72 65 Filename              'lorem.txt'
                 6D 2E 74 78
                 74

    016C 016F 0004 50 4B 05 06 END CENTRAL HEADER    06054B50 (101010256)
    0170 0171 0002 00 00       Number of this disk   0000 (0)
    0172 0173 0002 00 00       Central Dir Disk no   0000 (0)
    0174 0175 0002 01 00       Entries in this disk  0001 (1)
    0176 0177 0002 01 00       Total Entries         0001 (1)
    0178 017B 0004 37 00 00 00 Size of Central Dir   00000037 (55)
    017C 017F 0004 35 01 00 00 Offset to Central Dir 00000135 (309)
    0180 0181 0002 00 00       Comment Length        0000 (0)
    #
    # Done
```

Advanced Analysis

If you have a corrupt or non-standard zip file, particulatly one where the "Central Directory" metadata at the end of the file is absent/incomplete, you can use either the "--walk" option or the "--scan" option to search for any zip metadata that is still present in the file.

When either of these options is enabled, this program will bypass the initial step of reading the "Central Directory" at the end of the file and simply scan the zip file sequentially from the start of the file looking for zip metedata records. Although this can be error prone, for the most part it will find any zip file metadata that is still present in the file.

The difference between the two options is how aggressive the sequential scan is: "--walk" is optimistic, while "--scan" is pessimistic.

To understand the difference in more detail you need to know a bit about how zip file metadata is structured. Under the hood, a zip file uses a series of 4-byte signatures to

flag the start of a each of the metadata records it uses. When the "--walk" or the "--scan" option is enabled both work identically by scanning the file from the beginning looking for any the of these valid 4-byte metadata signatures. When a 4-byte signature is found both options will blindly assume that it has found a vald metadata record and display it.

"--walk"

The "--walk" option optimistically assumes that it has found a real zip metatada record and so starts the scan for the next record directly after the record it has just output.

"--scan"

The "--scan" option is pessimistic and assumes the 4-byte signature sequence may have been a false-positive, so before starting the scan for the next resord, it will rewind to the location in the file directly after the 4-byte sequecce it just processed. This means it will rescan data that has already been processed. For very lage zip files the "--scan" option can be really realy slow, so trying the "--walk" option first.

Important Note: If the zip file being processed contains one or more nested zip files, and the outer zip file uses the "STORE" compression method, the "--scan" option will display the zip metadata for both the outer & inner zip files.

Filename Encoding Issues

Sometimes when displaying the contents of a zip file the filenames (or comments) appear to be garbled. This section walks through the reasons and mitigations that can be applied to work around these issues.

Background

When zip files were first created in the 1980's, there was no Unicode or UTF-8. Issues around character set encoding interoperability were not a major concern.

Initially, the only official encoding supported in zip files was IBM Code Page 437 (AKA

"CP437"). As time went on users in locales where "CP437" wasn't appropriate stored filenames in the encoding native to their locale. If you were running a system that matched the locale of the zip file, all was well. If not, you had to post-process the filenames after unzipping the zip file.

Fast forward to the introduction of Unicode and UTF-8 encoding. The approach now used by all major zip implementations is to set the "Language encoding flag" (also known as "EFS") in the zip file metadata to signal that a filename/comment is encoded in UTF-8.

To ensure maximum interoperability when sharing zip files store 7-bit filenames as-is in the zip file. For anything else the "EFS" bit needs to be set and the filename is encoded in UTF-8. Although this rule is kept to for the most part, there are exceptions out in the wild.

Dealing with Encoding Errors

The most common filename encoding issue is where the "EFS" bit is not set and the filename is stored in a character set that doesnt't match the system encoding. This mostly impacts legacy zip files that predate the introduction of Unicode.

To deal with this issue you first need to know what encoding was used in the zip file. For example, if the filename is encoded in "ISO-8859-1" you can display the filenames using the "--encoding" option

```
zipdetails --encoding ISO-8859-1 myfile.zip
```

A less common variation of this is where the "EFS" bit is set, signalling that the filename will be encoded in UTF-8, but the filename is not encoded in UTF-8. To deal with this scenarion, use the "--no-language-encoding" option along with the "--encoding" option.

LIMITATIONS

The following zip file features are not supported by this program:

? Multi-part/Split/Spanned Zip Archives.

This program cannot give an overall report on the combined parts of a multi-part zip file.

The best you can do is run with either the "--scan" or "--walk" options against individual parts. Some will contains zipfile metadata which will be detected and some will only contain compressed payload data.

? Encrypted Central Directory

When pkzip Strong Encryption is enabled in a zip file this program can still parse most of the metadata in the zip file. The exception is when the "Central Directory" of a zip file is also encrypted. This program cannot parse any metadata from an encrypted "Central Directory".

? Corrupt Zip files

When "zipdetails" encounters a corrupt zip file, it will do one or more of the following

? Display details of the corruption and carry on

? Display details of the corruption and terminate

? Terminate with a generic message

Which of the above is output is dependent in the severity of the corruption.

TODO

JSON/YML Output

Output some of the zip file metadata as a JSON or YML document.

Corrupt Zip files

Although the detection and reporting of most of the common corruption use-cases is present in "zipdetails", there are likely to be other edge cases that need to be supported.

If you have a corrupt Zip file that isn't being processed properly, please report it (see "SUPPORT").

SUPPORT

General feedback/questions/bug reports should be sent to <https://github.com/pmqs/zipdetails/issues>.

SEE ALSO

The primary reference for Zip files is APPNOTE.TXT <https://pkware.cachefly.net/webdocs/casestudies/APPNOTE.TXT>.

An alternative reference is the Info-Zip appnote. This is available from <ftp://ftp.info-zip.org/pub/infozip/doc/>

For details of WinZip AES encryption see AES Encryption Information: Encryption Specification AE-1 and AE-2 <https://www.winzip.com/en/support/aes-encryption/>.

The "zipinfo" program that comes with the info-zip distribution (<http://www.info-zip.org/>) can also display details of the structure of a zip file.

AUTHOR

Paul Marquess pmqs@cpan.org.

COPYRIGHT