NAME

IO::Compress::Bzip2 - Write bzip2 files/buffers

SYNOPSIS

```
use IO::Compress::Bzip2 qw(bzip2 $Bzip2Error) ;

my $status = bzip2 $input => $output [,OPTS]
    or die "bzip2 failed: $Bzip2Error\n";

my $z = new IO::Compress::Bzip2 $output [,OPTS]
    or die "bzip2 failed: $Bzip2Error\n";

$z->print($string);
$z->printf($format, $string);
$z->write($string);
$z->syswrite($string [, $length, $offset]);
$z->flush();
$z->tell();
$z->eof();
$z->seek($position, $whence);
$z->binmode();
$z->fileno();
$z->opened();
$z->autoflush();
$z->input_line_number();
$z->newStream( [OPTS] );

$z->close() ;

$Bzip2Error ;

# IO::File mode

print $z $string;
printf $z $format, $string;
tell $z
eof $z
seek $z, $position, $whence
binmode $z
fileno $z
close $z ;
```

DESCRIPTION

This module provides a Perl interface that allows writing bzip2 compressed data to files or buffer.

For reading bzip2 files/buffers, see the companion module *IO::Uncompress::Bunzip2*.

Functional Interface

A top-level function, `bzip2`, is provided to carry out "one-shot" compression between buffers and/or files. For finer control over the compression process, see the *OO Interface* section.

```
use IO::Compress::Bzip2 qw(bzip2 $Bzip2Error) ;
```

```
    bzip2 $input => $output [,OPTS]
        or die "bzip2 failed: $Bzip2Error\n";
```

The functional interface needs Perl5.005 or better.

bzip2 $input => $output [, OPTS]

`bzip2` expects at least two parameters, `$input` and `$output`.

The $input parameter

The parameter, `$input`, is used to define the source of the uncompressed data.

It can take one of the following forms:

A filename

If the `$input` parameter is a simple scalar, it is assumed to be a filename. This file will be opened for reading and the input data will be read from it.

A filehandle

If the `$input` parameter is a filehandle, the input data will be read from it. The string '-' can be used as an alias for standard input.

A scalar reference

If `$input` is a scalar reference, the input data will be read from `$$input`.

An array reference

If `$input` is an array reference, each element in the array must be a filename.

The input data will be read from each file in turn.

The complete array will be walked to ensure that it only contains valid filenames before any data is compressed.

An Input FileGlob string

If `$input` is a string that is delimited by the characters "<" and ">" `bzip2` will assume that it is an *input fileglob string*. The input is the list of files that match the fileglob.

If the fileglob does not match any files ...

See *File::GlobMapper* for more details.

If the `$input` parameter is any other type, `undef` will be returned.

The $output parameter

The parameter `$output` is used to control the destination of the compressed data. This parameter can take one of these forms.

A filename

If the `$output` parameter is a simple scalar, it is assumed to be a filename. This file will be opened for writing and the compressed data will be written to it.

A filehandle

If the `$output` parameter is a filehandle, the compressed data will be written to it. The string '-' can be used as an alias for standard output.

A scalar reference

If `$output` is a scalar reference, the compressed data will be stored in `$$output`.

An Array Reference

If `$output` is an array reference, the compressed data will be pushed onto the array.

An Output FileGlob

If `$output` is a string that is delimited by the characters "<" and ">" `bzip2` will assume that it is an *output fileglob string*. The output is the list of files that match the fileglob.

When `$output` is an fileglob string, `$input` must also be a fileglob string. Anything else is an error.

If the `$output` parameter is any other type, `undef` will be returned.

Notes

When `$input` maps to multiple files/buffers and `$output` is a single file/buffer the input files/buffers will be stored in `$output` as a concatenated series of compressed data streams.

Optional Parameters

Unless specified below, the optional parameters for `bzip2`, `OPTS`, are the same as those used with the OO interface defined in the *Constructor Options* section below.

`AutoClose => 0|1`

This option applies to any input or output data streams to `bzip2` that are filehandles.

If `AutoClose` is specified, and the value is true, it will result in all input and/or output filehandles being closed once `bzip2` has completed.

This parameter defaults to 0.

`BinModeIn => 0|1`

When reading from a file or filehandle, set `binmode` before reading.

Defaults to 0.

`Append => 0|1`

TODO

Examples

To read the contents of the file `file1.txt` and write the compressed data to the file `file1.txt.bz2`.

```
    use strict ;
    use warnings ;
    use IO::Compress::Bzip2 qw(bzip2 $Bzip2Error) ;

    my $input = "file1.txt";
    bzip2 $input => "$input.bz2"
        or die "bzip2 failed: $Bzip2Error\n";
```

To read from an existing Perl filehandle, `$input`, and write the compressed data to a buffer, `$buffer`.

```
    use strict ;
    use warnings ;
    use IO::Compress::Bzip2 qw(bzip2 $Bzip2Error) ;
    use IO::File ;

    my $input = new IO::File "<file1.txt"
        or die "Cannot open 'file1.txt': $!\n" ;
    my $buffer ;
    bzip2 $input => \$buffer
        or die "bzip2 failed: $Bzip2Error\n";
```

To compress all files in the directory "/my/home" that match "*.txt" and store the compressed data in the same directory

```
use strict ;
use warnings ;
use IO::Compress::Bzip2 qw(bzip2 $Bzip2Error) ;

bzip2 '</my/home/*.txt>' => '<*.bz2>'
    or die "bzip2 failed: $Bzip2Error\n";
```

and if you want to compress each file one at a time, this will do the trick

```
use strict ;
use warnings ;
use IO::Compress::Bzip2 qw(bzip2 $Bzip2Error) ;

for my $input ( glob "/my/home/*.txt" )
{
    my $output = "$input.bz2" ;
    bzip2 $input => $output
        or die "Error compressing '$input': $Bzip2Error\n";
}
```

OO Interface

Constructor

The format of the constructor for `IO::Compress::Bzip2` is shown below

```
my $z = new IO::Compress::Bzip2 $output [,OPTS]
    or die "IO::Compress::Bzip2 failed: $Bzip2Error\n";
```

It returns an `IO::Compress::Bzip2` object on success and undef on failure. The variable `$Bzip2Error` will contain an error message on failure.

If you are running Perl 5.005 or better the object, `$z`, returned from IO::Compress::Bzip2 can be used exactly like an *IO::File* filehandle. This means that all normal output file operations can be carried out with `$z`. For example, to write to a compressed file/buffer you can use either of these forms

```
$z->print("hello world\n");
print $z "hello world\n";
```

The mandatory parameter `$output` is used to control the destination of the compressed data. This parameter can take one of these forms.

A filename

If the `$output` parameter is a simple scalar, it is assumed to be a filename. This file will be opened for writing and the compressed data will be written to it.

A filehandle

If the `$output` parameter is a filehandle, the compressed data will be written to it. The string '-' can be used as an alias for standard output.

A scalar reference

If `$output` is a scalar reference, the compressed data will be stored in `$$output`.

If the `$output` parameter is any other type, `IO::Compress::Bzip2`::new will return undef.

Constructor Options

OPTS is any combination of the following options:

`AutoClose => 0|1`

This option is only valid when the `$output` parameter is a filehandle. If specified, and the value is true, it will result in the `$output` being closed once either the `close` method is called or the `IO::Compress::Bzip2` object is destroyed.

This parameter defaults to 0.

`Append => 0|1`

Opens `$output` in append mode.

The behaviour of this option is dependent on the type of `$output`.

* A Buffer

If `$output` is a buffer and `Append` is enabled, all compressed data will be append to the end if `$output`. Otherwise `$output` will be cleared before any data is written to it.

* A Filename

If `$output` is a filename and `Append` is enabled, the file will be opened in append mode. Otherwise the contents of the file, if any, will be truncated before any compressed data is written to it.

* A Filehandle

If `$output` is a filehandle, the file pointer will be positioned to the end of the file via a call to `seek` before any compressed data is written to it. Otherwise the file pointer will not be moved.

This parameter defaults to 0.

`BlockSize100K => number`

Specify the number of 100K blocks bzip2 uses during compression.

Valid values are from 1 to 9, where 9 is best compression.

The default is 1.

`WorkFactor => number`

Specifies how much effort bzip2 should take before resorting to a slower fallback compression algorithm.

Valid values range from 0 to 250, where 0 means use the default value 30.

The default is 0.

`Strict => 0|1`

This is a placeholder option.

Examples

TODO

Methods

print

Usage is

```
$z->print($data)
print $z $data
```

Compresses and outputs the contents of the `$data` parameter. This has the same behaviour as the `print` built-in.

Returns true if successful.

printf

Usage is

```
$z->printf($format, $data)
printf $z $format, $data
```

Compresses and outputs the contents of the `$data` parameter.

Returns true if successful.

syswrite

Usage is

```
$z->syswrite $data
$z->syswrite $data, $length
$z->syswrite $data, $length, $offset
```

Compresses and outputs the contents of the `$data` parameter.

Returns the number of uncompressed bytes written, or `undef` if unsuccessful.

write

Usage is

```
$z->write $data
$z->write $data, $length
$z->write $data, $length, $offset
```

Compresses and outputs the contents of the `$data` parameter.

Returns the number of uncompressed bytes written, or `undef` if unsuccessful.

flush

Usage is

```
$z->flush;
```

Flushes any pending compressed data to the output file/buffer.

TODO

Returns true on success.

tell

Usage is

```
$z->tell()
tell $z
```

Returns the uncompressed file offset.

eof

Usage is

```
$z->eof();
eof($z);
```

Returns true if the `close` method has been called.

seek

```
$z->seek($position, $whence);
seek($z, $position, $whence);
```

Provides a sub-set of the `seek` functionality, with the restriction that it is only legal to seek forward in the output file/buffer. It is a fatal error to attempt to seek backward.

Empty parts of the file/buffer will have NULL (0x00) bytes written to them.

The `$whence` parameter takes one the usual values, namely SEEK_SET, SEEK_CUR or SEEK_END.

Returns 1 on success, 0 on failure.

binmode

Usage is

```
$z->binmode
binmode $z ;
```

This is a noop provided for completeness.

opened

```
$z->opened()
```

Returns true if the object currently refers to a opened file/buffer.

autoflush

```
my $prev = $z->autoflush()
my $prev = $z->autoflush(EXPR)
```

If the `$z` object is associated with a file or a filehandle, this method returns the current autoflush setting for the underlying filehandle. If EXPR is present, and is non-zero, it will enable flushing after every write/print operation.

If `$z` is associated with a buffer, this method has no effect and always returns `undef`.

Note that the special variable `$|` **cannot** be used to set or retrieve the autoflush setting.

input_line_number

```
$z->input_line_number()
$z->input_line_number(EXPR)
```

This method always returns `undef` when compressing.

fileno

```
$z->fileno()
fileno($z)
```

If the $z object is associated with a file or a filehandle, fileno will return the underlying file descriptor. Once the close method is called fileno will return undef.

If the $z object is is associated with a buffer, this method will return undef.

close

```
$z->close() ;
close $z ;
```

Flushes any pending compressed data and then closes the output file/buffer.

For most versions of Perl this method will be automatically invoked if the IO::Compress::Bzip2 object is destroyed (either explicitly or by the variable with the reference to the object going out of scope). The exceptions are Perl versions 5.005 through 5.00504 and 5.8.0. In these cases, the close method will be called automatically, but not until global destruction of all live objects when the program is terminating.

Therefore, if you want your scripts to be able to run on all versions of Perl, you should call close explicitly and not rely on automatic closing.

Returns true on success, otherwise 0.

If the AutoClose option has been enabled when the IO::Compress::Bzip2 object was created, and the object is associated with a file, the underlying file will also be closed.

newStream([OPTS])

Usage is

```
$z->newStream( [OPTS] )
```

Closes the current compressed data stream and starts a new one.

OPTS consists of any of the the options that are available when creating the $z object.

See the *Constructor Options* section for more details.

Importing

No symbolic constants are required by this IO::Compress::Bzip2 at present.

:all

Imports bzip2 and $Bzip2Error. Same as doing this

```
use IO::Compress::Bzip2 qw(bzip2 $Bzip2Error) ;
```

EXAMPLES

Apache::GZip Revisited

See *IO::Compress::Bzip2::FAQ*

Working with Net::FTP

See *IO::Compress::Bzip2::FAQ*

SEE ALSO

Compress::Zlib, *IO::Compress::Gzip*, *IO::Uncompress::Gunzip*, *IO::Compress::Deflate*, *IO::Uncompress::Inflate*, *IO::Compress::RawDeflate*, *IO::Uncompress::RawInflate*, *IO::Uncompress::Bunzip2*, *IO::Compress::Lzop*, *IO::Uncompress::UnLzop*, *IO::Compress::Lzf*,

IO::Uncompress::UnLzf, *IO::Uncompress::AnyInflate*, *IO::Uncompress::AnyUncompress*

Compress::Zlib::FAQ

File::GlobMapper, *Archive::Zip*, *Archive::Tar*, *IO::Zlib*

The primary site for the bzip2 program is *http://www.bzip.org*.

See the module *Compress::Bzip2*

AUTHOR

This module was written by Paul Marquess, *pmqs@cpan.org*.

MODIFICATION HISTORY

See the Changes file.

COPYRIGHT AND LICENSE