NAME

Dpkg::Compression::FileHandle - object dealing transparently with file compression

SYNOPSIS

```
use Dpkg::Compression::FileHandle;

$fh = Dpkg::Compression::FileHandle->new(filename => 'sample.gz');
print $fh "Something\n";
close $fh;

$fh = Dpkg::Compression::FileHandle->new();
open($fh, '>', 'sample.bz2');
print $fh "Something\n";
close $fh;

$fh = Dpkg::Compression::FileHandle->new();
$fh->open('sample.xz', 'w');
$fh->print("Something\n");
$fh->close();

$fh = Dpkg::Compression::FileHandle->new(filename => 'sample.gz');
my @lines = <$fh>;
close $fh;

$fh = Dpkg::Compression::FileHandle->new();
open($fh, '<', 'sample.bz2');
my @lines = <$fh>;
close $fh;

$fh = Dpkg::Compression::FileHandle->new();
$fh->open('sample.xz', 'r');
my @lines = $fh->getlines();
$fh->close();
```

DESCRIPTION

Dpkg::Compression::FileHandle is an object that can be used like any filehandle and that deals transparently with compressed files. By default, the compression scheme is guessed from the filename but you can override this behaviour with the method `set_compression`.

If you don't open the file explicitly, it will be auto-opened on the first read or write operation based on the filename set at creation time (or later with the `set_filename` method).

Once a file has been opened, the filehandle must be closed before being able to open another file.

STANDARD FUNCTIONS

The standard functions acting on filehandles should accept a Dpkg::Compression::FileHandle object transparently including `open` (only when using the variant with 3 parameters), `close`, `binmode`, `eof`, `fileno`, `getc`, `print`, `printf`, `read`, `sysread`, `say`, `write`, `syswrite`, `seek`, `sysseek`, `tell`.

Note however that `seek` and `sysseek` will only work on uncompressed files as compressed files are really pipes to the compressor programs and you can't seek on a pipe.

FileHandle METHODS

The object inherits from FileHandle so all methods that work on this object should work for Dpkg::Compression::FileHandle too. There may be exceptions though.

PUBLIC METHODS

my `$fh` = Dpkg::Compression::FileHandle->new(%opts)

Creates a new filehandle supporting on-the-fly compression/decompression. Supported options are ''filename'', ''compression'', ''compression_level'' (see respective set_* functions) and ''add_comp_ext''. If ''add_comp_ext'' evaluates to true, then the extension corresponding to the selected compression scheme is automatically added to the recorded filename. It's obviously incompatible with automatic detection of the compression method.

`$fh`->ensure_open($mode, `%opts`)

Ensure the file is opened in the requested mode (''r'' for read and ''w'' for write). The options are passed down to the compressor's *spawn()* call, if one is used. Opens the file with the recorded filename if needed. If the file is already open but not in the requested mode, then it errors out.

`$fh`->set_compression($comp)

Defines the compression method used. `$comp` should one of the methods supported by **Dpkg::Compression** or ''none'' or ''auto''. ''none'' indicates that the file is uncompressed and ''auto'' indicates that the method must be guessed based on the filename extension used.

`$fh`->set_compression_level($level)

Indicate the desired compression level. It should be a value accepted by the function `compression_is_valid_level` of **Dpkg::Compression**

`$fh`->set_filename($name, [$add_comp_ext])

Use `$name` as filename when the file must be opened/created. If `$add_comp_ext` is passed, it indicates whether the default extension of the compression method must be automatically added to the filename (or not).

my `$file` = `$fh`->*get_filename()*

Returns the filename that would be used when the filehandle must be opened (both in read and write mode). This function errors out if ''add_comp_ext'' is enabled while the compression method is set to ''auto''. The returned filename includes the extension of the compression method if ''add_comp_ext'' is enabled.

`$ret` = `$fh`->*use_compression()*

Returns ''0'' if no compression is used and the compression method used otherwise. If the compression is set to ''auto'', the value returned depends on the extension of the filename obtained with the **get_filename** method.

my `$real_fh` = `$fh`->*get_filehandle()*

Returns the real underlying filehandle. Useful if you want to pass it along in a derived object.

DERIVED OBJECTS

If you want to create an object that inherits from Dpkg::Compression::FileHandle you must be aware that the object is a reference to a GLOB that is returned by *Symbol::gensym()* and as such it's not a HASH.

You can store internal data in a hash but you have to use `*$self-{...}>` to access the associated hash like in the example below:

```
sub set_option {
my ($self, $value) = @_;
*$self->{option} = $value;
}
```

CHANGES

Version 1.01

New argument: `$fh`->*ensure_open()* accepts an `%opts` argument.

Version 1.00

Mark the module as public.

AUTHOR

Raphal Hertzog <hertzog@debian.org>