

THIS FILE is almost entirely based upon code by Jean-loup Gailly and Mark Adler. It has been modified by Lucian Wischik. The modifications were: incorporate the bugfixes of 1.1.4, allow unzipping to/from handles/pipes/files/memory, encryption, unicode, a windowsish api, and putting everything into a single .cpp file. The original code may be found at <http://www.gzip.org/zlib/> The original copyright text follows.

zlib.h -- interface of the 'zlib' general purpose compression library version 1.1.3, July 9th, 1998

Copyright (C) 1995-1998 Jean-loup Gailly and Mark Adler

This software is provided 'as-is', without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.
2. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.
3. This notice may not be removed or altered from any source distribution.

Jean-loup Gailly	Mark Adler
jloup@gzip.org	madler@alumni.caltech.edu

The data format used by the zlib library is described by RFCs (Request for Comments) 1950 to 1952 in the files <ftp://ds.internic.net/rfc/rfc1950.txt> (zlib format), [rfc1951.txt](#) (deflate format) and [rfc1952.txt](#) (gzip format).

The 'zlib' compression library provides in-memory compression and decompression functions, including integrity checks of the uncompressed data. This version of the library supports only one compression method (deflation) but other algorithms will be added later and will have the same stream interface.

Compression can be done in a single step if the buffers are large enough (for example if an input file is `mmap`'ed), or can be done by repeated calls of the compression function. In the latter case, the application must provide more

input and/or consume the output (providing more output space) before each call.

The library also supports reading and writing files in gzip (.gz) format with an interface similar to that of stdio.

The library does not install any signal handler. The decoder checks the consistency of the compressed data, so the library should never crash even in case of corrupted input.