
Copyright (c) 2009, 2010, 2013-2015 by the Brotli Authors.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

This is a plugin based on the brotli library released under the MIT license (see above).

The scope of this library is to compress/decompress brotli archives and buffers on Android, iOS, Windows, OSX & Linux.

The examples require some files that reside in the StreamingAssets folder. They are there just for testing purposes. In your final projects make sure you delete them so that you don't increase your build size.

The ios libraries are compiled as universal amd bitcode enabled. That means that they will support 32 and 64 bit builds.

TvOS is supported.

(non-bitcode enabled iOS plugins are provided. If you are creating a non bitcode enabled project please use these provided plugins!)

OSX bundle is compiled now as 64-bit only, since Apple store requires it.

The Windows and Linux libraries are compiled for x86 and x86_64 build modes.

The Android lib is compiled for armeabi-v7a, x86 and arm64-v8a.

FEATURES:

Fast brotli compression and decompression with a clean and simple interface. Very easy to use.

- compress a file into a brotli file format.
- decompress a brotli file.
- compress a buffer into the brotli format.
- decompress a brotli compressed buffer into a buffer.

(The plugin provides a solution to store the uncompressed size of a compressed buffer in its footer.)

- Linux, iOS, Android, MacOSX can treat buffers as files. That means if you have a file in `www.bytes` you can perform operations directly on the buffer.

For Android this is very useful since you can decompress from Streaming Assets without copying to Persistent data path.

!!! If you want to use only the brotli plugin, please delete all the other plugins in their respective folders or use the single packages from the `_plugin_packages` folder!!!.

INSTRUCTIONS:

If you want to run a small example, compile and run the `testScene`.
It will download a small `tif` file and it will perform all the functions the lib provides.

See the `brotli.cs` file for more comments and error codes.

In your project include in the `Plugins` folder the plugins you want to use and the `brotli.cs` file and call the appropriate functions as described below and shown in the demo scene.

FUNCTIONS:

```
int compressFile(string inFile, string outFile, int[] proc, int quality = 9, int lgwin = 19, int lgblock = 0, int mode = 0);
```

Compress a file to brotli format.

Full paths to the files should be provided.

InFile : The input file

outFile: The output file

proc : A single item referenced int array to provide progress of compression

quality: (0 - 11) quality of compression (0 = faster/bigger - 11 = slower/smaller).

lgwin : Base 2 logarithm of the sliding window size. Range is 10 to 24. (10 - 24) memory used for compression (higher numbers use more ram)

lgblock: 0 for auto or 16-24. Base 2 logarithm of the maximum input block size. Range is 16 to 24. If set to 0, the value will be set based on the quality.

mode : (0 - 2) 0 = default, 1 = utf8 text, 2 = woff 2.0 font

error codes:

- 1 : OK
- 1 : compression failed
- 2 : not enough memory
- 3 : could not close in file
- 4 : could not close out file

```
int decompressFile(string inFile, string outFile, int[] proc, byte[] FileBuffer = null);
```

Decompress a brotli file.

Full paths to the files should be provided.

InFile : The input file
outFile : The output file
proc : A single item referenced int array to provide progress of decompression
FileBuffer : A buffer that holds a brotli file. When assigned the function will read from this buffer and will ignore the filePath. (Linux, iOS, Android, MacOSX)
returns : 1 on success.

error codes : 1 : OK
-1 : failed to write output
-2 : corrupt input
-3 : could not close in file
-4 : could not close out file

```
int getDecodedSize(byte[] inBuffer);
```

Get the uncompressed size of a brotli buffer. This will work only on small buffers with one metablock.

inBuffer: the input buffer that stores a brotli compressed buffer.

```
bool compressBuffer(byte[] inBuffer, ref byte[] outBuffer, int[] proc, bool includeSize = false, int quality = 9, int lgwin = 19, int lgblock = 0, int mode = 0);
```

Compress a byte buffer in brotli format.

Returns true on success.

inBuffer : the uncompressed buffer.
outBuffer : a **referenced** buffer that will store the compressed data. (it should be large enough to store it.)
proc : A single item referenced int array to provide progress of compression
includeSize : include the uncompressed size of the buffer in the resulted compressed one because brotli does not support it for larger than 1 metablock.
quality : (0 - 11) quality of compression (0 = faster/bigger - 11 = slower/smaller).
lgwin : (10 - 24) memory used for compression (higher numbers use more ram). Base 2 logarithm of the sliding window size. Range is 10 to 24.
lgblock : 0 for auto or 16-24. Base 2 logarithm of the maximum input block size. Range is 16 to 24. If set to 0, the value will be set based on the quality.
Mode : (0 - 2) 0 = default, 1 = utf8 text, 2 = woff 2.0 font

```
byte[] compressBuffer(byte[] inBuffer, int[] proc, bool includeSize = false, int quality = 9, int lgwin = 19, int lgblock = 0, int mode = 0);
```

Same as above only this function returns a new created buffer with the compressed data.

```
int compressBuffer(byte[] inBuffer, byte[] outBuffer, int[] proc, int quality = 9, int lgwin = 19,
int lgblock = 0, int mode = 0);
```

*Same as `bool compressBuffer` , only this time the compressed buffer is written in a **fixed** size buffer.*

Returns the compressed size in bytes.

```
bool decompressBuffer(byte[] inBuffer, ref byte[] outBuffer, bool useFooter = false);
```

*Decompress a brotli compressed buffer to a referenced buffer.
Returns true on success.*

inBuffer : the brotli compressed buffer
outBuffer: a referenced buffer that will be resized to store the uncompressed data.
useFooter: if the input Buffer has the uncompressed size info.

```
byte[] decompressBuffer(byte[] inBuffer, bool useFooter = false);
```

Same as above only this time the uncompressed data is returned in a new created buffer.

```
int decompressBuffer(byte[] inBuffer, byte[] outBuffer, bool useFooter = false);
```

Same as above only the decompressed data will be stored in a fixed size outBuffer.

Make sure the fixed buffer is big enough to store the data.

Returns: uncompressed size in bytes.

[Android, iOS, Linux, MacOSX only]

```
int setFilePermissions(string filePath, string _user, string _group, string _other);
```

Sets permissions of a file in user, group, other.

Each string should contain any or all chars of "rwx".

Returns 0 on success.

SUPPORT:

For any questions, problems and suggestions please use this email address:
elias_t@yahoo.com

forum: <http://forum.unity3d.com/threads/7zip-lzma-and-zip-native-multiplatform->