





[Discover Packages](#) > [Standard library](#) > [compress](#) > [lzw](#) **lzw**

package

standard library

Version: [go1.20.1](#) **Latest** | Published: Feb 14, 2023 | License: [BSD-3-Clause](#) | Imports: [4](#) |Imported by: [567](#)**Details**

- ✓ Valid [go.mod](#) file 
- ✓ Redistributable license 
- ✓ Tagged version 
- ✓ Stable version 

[Learn more](#)**Repository**cs.opensource.google/go/go**Links** [Report a Vulnerability](#) Documentation <> **Documentation****Overview**

Package `lzw` implements the Lempel-Ziv-Welch compressed data format, described in T. A. Welch, “A Technique for High-Performance Data Compression”, *Computer*, 17(6) (June 1984), pp 8-19.

In particular, it implements LZW as used by the GIF and PDF file formats, which means variable-width codes up to 12 bits and the first two non-literal codes are a clear code and an EOF code.

The TIFF file format uses a similar but incompatible version of the LZW algorithm. See the golang.org/x/image/tiff/lzw package for an implementation.

Index

```
func NewReader(r io.Reader, order Order, litWidth int) io.ReadCloser
```

```
func NewWriter(w io.Writer, order Order, litWidth int) io.WriteCloser
```

```
type Order
```

```
type Reader
```

```
func (r *Reader) Close() error
```

```
func (r *Reader) Read(b []byte) (int, error)
```

```
func (r *Reader) Reset(src io.Reader, order Order, litWidth int)
```

```
type Writer
```

```
func (w *Writer) Close() error
```

```
func (w *Writer) Reset(dst io.Writer, order Order, litWidth int)
```

```
func (w *Writer) Write(p []byte) (n int, err error)
```

Constants

This section is empty.

Variables

This section is empty.

Functions

func NewReader

```
func NewReader(r io.Reader, order Order, litWidth int) io.ReadCloser
```

NewReader creates a new io.ReadCloser. Reads from the returned io.ReadCloser read and decompress data from r. If r does not also implement io.ByteReader, the decompressor may read more data than necessary from r. It is the caller's responsibility to call Close on the ReadCloser when finished reading. The number of bits to use for literal codes, litWidth, must be in the range [2,8] and is typically 8. It must equal the litWidth used during compression.

It is guaranteed that the underlying type of the returned io.ReadCloser is a *Reader.

func NewWriter

```
func NewWriter(w io.Writer, order Order, litWidth int) io.WriteCloser
```

NewWriter creates a new io.WriteCloser. Writes to the returned io.WriteCloser are compressed and written to w. It is the caller's responsibility to call Close on the WriteCloser when finished writing. The number of bits to use for literal codes, litWidth, must be in the range [2,8] and is typically 8. Input bytes must be less than 1<<litWidth.

It is guaranteed that the underlying type of the returned io.WriteCloser is a *Writer.

Types

type Order

```
type Order int
```

Order specifies the bit ordering in an LZW data stream.

```
const (  
    // LSB means Least Significant Bits first, as used in the GIF file format.  
    LSB Order = iota  
    // MSB means Most Significant Bits first, as used in the TIFF and PDF  
    // file formats.  
    MSB  
)
```

type Reader

added in go1.17

```
type Reader struct {  
    // contains filtered or unexported fields  
}
```

Reader is an `io.Reader` which can be used to read compressed data in the LZW format.

func (*Reader) Close

added in go1.17

```
func (r *Reader) Close() error
```

Close closes the Reader and returns an error for any future read operation. It does not close the underlying `io.Reader`.

func (*Reader) Read

added in go1.17

```
func (r *Reader) Read(b []byte) (int, error)
```

Read implements `io.Reader`, reading uncompressed bytes from its underlying Reader.

func (*Reader) Reset

added in go1.17

```
func (r *Reader) Reset(src io.Reader, order Order, litWidth int)
```

Reset clears the Reader's state and allows it to be reused again as a new Reader.

type Writer

added in go1.17

```
type Writer struct {  
    // contains filtered or unexported fields  
}
```

Writer is an LZW compressor. It writes the compressed form of the data to an underlying writer (see `NewWriter`).

func (*Writer) Close

added in go1.17

```
func (w *Writer) Close() error
```

Close closes the Writer, flushing any pending output. It does not close w's underlying writer.

func (*Writer) Reset

added in go1.17

```
func (w *Writer) Reset(dst io.Writer, order Order, litWidth int)
```

Reset clears the Writer's state and allows it to be reused again as a new Writer.

func (*Writer) Write

added in go1.17

```
func (w *Writer) Write(p []byte) (n int, err error)
```

Write writes a compressed representation of p to w's underlying writer.

Source Files

[View all](#) 

[reader.go](#)

[writer.go](#)

Why Go

[Use Cases](#)

[Case Studies](#)

Get Started

[Playground](#)

[Tour](#)

[Stack Overflow](#)

[Help](#)

Packages

[Standard Library](#)

[About Go Packages](#)

About

[Download](#)

[Blog](#)

[Issue Tracker](#)

[Release Notes](#)

[Brand Guidelines](#)

[Code of Conduct](#)

Connect

[Twitter](#)

[GitHub](#)

[Slack](#)

[r/golang](#)

[Meetup](#)

[Golang Weekly](#)

[Copyright](#)

[Terms of Service](#)

[Privacy Policy](#)

[Report an Issue](#)



Google