errors

package standard library

Version: go1.20.1 Latest | Published: Feb 14, 2023 | License: BSD-3-Clause | Imports: 1 | Imported by: 782,705

Details ✓ Valid go.mod file ✓ Redistributable license ✓ Tagged version ✓ Stable version

Learn more

Repository cs.opensource.google/go/go

Links Report a Vulnerability

Documentation

Overview

Package errors implements functions to manipulate errors.

The New function creates errors whose only content is a text message.

An error e wraps another error if e's type has one of the methods

```
Unwrap() error
Unwrap() []error
```

If e.Unwrap() returns a non-nil error w or a slice containing w, then we say that e wraps w. A nil error returned from e.Unwrap() indicates that e does not wrap any error. It is invalid for an Unwrap method to return an []error containing a nil error value.

An easy way to create wrapped errors is to call fmt.Errorf and apply the %w verb to the error argument:

```
wrapsErr := fmt.Errorf("... %w ...", ..., err, ...)
```

Successive unwrapping of an error creates a tree. The Is and As functions inspect an error's tree by examining first the error itself followed by the tree of each of its children in turn (pre-order, depth-first traversal).

Is examines the tree of its first argument looking for an error that matches the second. It reports whether it finds a match. It should be used in preference to simple equality checks:

```
if errors.Is(err, fs.ErrExist)
```

is preferable to

```
if err == fs.ErrExist
```

because the former will succeed if err wraps fs.ErrExist.

As examines the tree of its first argument looking for an error that can be assigned to its second argument, which must be a pointer. If it succeeds, it performs the assignment and returns true. Otherwise, it returns false. The form

```
var perr *fs.PathError
if errors.As(err, &perr) {
    fmt.Println(perr.Path)
}
```

is preferable to

```
if perr, ok := err.(*fs.PathError); ok {
    fmt.Println(perr.Path)
}
```

because the former will succeed if err wraps an *fs.PathError.

▶ Example

Index

Examples

Constants

This section is empty.

Variables

This section is empty.

Functions

func As

added in go1.13

```
func As(err error, target any) bool
```

As finds the first error in err's tree that matches target, and if one is found, sets target to that error value and returns true. Otherwise, it returns false.

The tree consists of err itself, followed by the errors obtained by repeatedly calling Unwrap. When err wraps multiple errors, As examines err followed by a depth-first traversal of its children.

An error matches target if the error's concrete value is assignable to the value pointed to by target, or if the error has a method As(interface{}) bool such that As(target) returns true. In the latter case, the As method is responsible for setting target.

An error type might provide an As method so it can be treated as if it were a different error type.

As panics if target is not a non-nil pointer to either a type that implements error, or to any interface type.

▶ Example

func Is

added in go1.13

```
func Is(err, target error) bool
```

Is reports whether any error in err's tree matches target.

The tree consists of err itself, followed by the errors obtained by repeatedly calling Unwrap. When err wraps multiple errors, Is examines err followed by a depth-first traversal of its children.

An error is considered to match a target if it is equal to that target or if it implements a method Is(error) bool such that Is(target) returns true.

An error type might provide an Is method so it can be treated as equivalent to an existing error. For example, if MyError defines

```
func (m MyError) Is(target error) bool { return target == fs.ErrExist }
```

then Is(MyError{}, fs.ErrExist) returns true. See syscall.Errno.Is for an example in the standard library. An Is method should only shallowly compare err and the target and not call Unwrap on either.

▶ Example

func Join

added in go1.20

```
func Join(errs ...error) error
```

Join returns an error that wraps the given errors. Any nil error values are discarded. Join returns nil if errs contains no non-nil values. The error formats as the concatenation of the strings obtained by calling the Error method of each element of errs, with a newline between each string.

▶ Example

func New

```
func New(text string) error
```

New returns an error that formats as the given text. Each call to New returns a distinct error value even if the text is identical.

▶ Example

▶ Example (Errorf)

func Unwrap

added in go1.13

```
func Unwrap(err error) error
```

Unwrap returns the result of calling the Unwrap method on err, if err's type contains an Unwrap method returning error. Otherwise, Unwrap returns nil.

Unwrap returns nil if the Unwrap method returns []error.

▶ Example

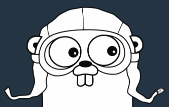
Types

This section is empty.

Source Files

View all

errors.go join.go wrap.go



Google