path

package standard library

Version: go1.20.1 Latest | Published: Feb 14, 2023 | License: BSD-3-Clause | Imports: 3 | Imported by: 174,120

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 path implements utility routines for manipulating slash-separated paths.

The path package should only be used for paths separated by forward slashes, such as the paths in URLs. This package does not deal with Windows paths with drive letters or backslashes; to manipulate operating system paths, use the path/filepath package.

Index

Examples

Constants

This section is empty.

Variables

View Source

```
var ErrBadPattern = errors.New("syntax error in pattern")
```

ErrBadPattern indicates a pattern was malformed.

Functions

func Base

```
func Base(path string) string
```

Base returns the last element of path. Trailing slashes are removed before extracting the last element. If the path is empty, Base returns ".". If the path consists entirely of slashes, Base returns "/".

▶ Example

func Clean

```
func Clean(path string) string
```

Clean returns the shortest path name equivalent to path by purely lexical processing. It applies the following rules iteratively until no further processing can be done:

1. Replace multiple slashes with a single slash.
2. Eliminate each . path name element (the current directory).
3. Eliminate each inner .. path name element (the parent directory) along with the non-.. element that precedes it.
4. Eliminate .. elements that begin a rooted path: that is, replace "/.." by "/" at the beginning of a path.

The returned path ends in a slash only if it is the root "/".

If the result of this process is an empty string, Clean returns the string ".".

See also Rob Pike, “Lexical File Names in Plan 9 or Getting Dot-Dot Right,” https://9p.io/sys/doc/lexnames.html

▶ Example

func Dir

```
func Dir(path string) string
```

Dir returns all but the last element of path, typically the path's directory. After dropping the final element using Split, the path is Cleaned and trailing slashes are removed. If the path is empty, Dir returns ".". If the path consists entirely of slashes followed by non-slash bytes, Dir returns a single slash. In any other case, the returned path does not end in a slash.

▶ Example

func Ext

```
func Ext(path string) string
```

Ext returns the file name extension used by path. The extension is the suffix beginning at the final dot in the final slash-separated element of path; it is empty if there is no dot.

▶ Example

func IsAbs

```
func IsAbs(path string) bool
```

IsAbs reports whether the path is absolute.

▶ Example

func Join

```
func Join(elem ...string) string
```

Join joins any number of path elements into a single path, separating them with slashes. Empty elements are ignored. The result is Cleaned. However, if the argument list is empty or all its elements are empty, Join returns an empty string.

▶ Example

func Match

```
func Match(pattern, name string) (matched bool, err error)
```

Match reports whether name matches the shell pattern. The pattern syntax is:

```
pattern:
    { term }
term:
    '*'         matches any sequence of non-/ characters
    '?'         matches any single non-/ character
    '[' [ '^' ] { character-range } ']'
                character class (must be non-empty)
    c           matches character c (c != '*', '?', '\\', '[')
    '\\' c      matches character c

character-range:
    c           matches character c (c != '\\', '-', ']')
    '\\' c      matches character c
    lo '-' hi   matches character c for lo <= c <= hi
```

Match requires pattern to match all of name, not just a substring. The only possible returned error is ErrBadPattern, when pattern is malformed.

▶ Example

func Split

```
func Split(path string) (dir, file string)
```

Split splits path immediately following the final slash, separating it into a directory and file name component. If there is no slash in path, Split returns an empty dir and file set to path. The returned values have the property that path = dir+file.

▶ Example

Types

This section is empty.

Source Files

View all

match.go   path.go

Directories

filepath

Package filepath implements utility routines for manipulating filename paths in a way compatible with the target operating system-defined file paths.

Google