go files Cheat Sheet

by cizixs (cizixs) via cheatography.com/43449/cs/12942/

Basic Operations

create empty file	`newFile, err := os.Cre ate ("te st.t xt ")`
truncate a file	`err := os.Tru nca te( " tes t.t xt", 100)`
get file info	`fileInfo, err := os.Sta te( " tes t.t xt")`
rename a file	`err := os.Ren ame (ol dPath, newPath)`
delete a file	`err := os.Rem ove ("te st.t xt ")`
open a file for reading	`file, err := os.Ope n("t est.tx t")`
open a file	`file, err := os.Ope n("t est.tx t", os.O_A PPEND, 0600)`
close a file	`err := file.C lose()`
change file permision	`err := os.Chm od( " tes t.t xt", 0777)`
change file ownership	`err := os.Cho wn( " tes t.t xt", os.Get uid(), os.Get gid())`
change file timestamps	`err := os.Cht ime s("t est.tx t", lastAc ces s, lastMo di f yTime)`

file open flag

`os.O_R DONLY`	open the file read only
`os.O_W RONLY`	open the file write only
`os.O_RDWR`	open the file read write
`os.O_A PPEND`	append data to the file when writing
`os.O_C REATE`	create a new file if none exists
`os.O_EXCL`	used with `O_CREATE`, file must not exist
`os.O_SYNC`	open for synchronous I/O
`O_TRUNC`	if possible, truncate file when opened

When opening file with `os.Ope nFile`, flags control how the file behaves.

Hard Link & Symbol Link

create a hard link	`err := os.Lin k("t est.tx t", " tes t_c opy.t x t")`
create a symbol link	`err := os.Sym lin k("t est.tx t", " tes t_s y m.t xt ")`
get link file info	`fileInfo, err := os.Lst at( " tes t_s ym.t xt ")`
change link file owner	`err := os.Lch own ("te st_ sym.tx t", uid, gi d)`
read a link	`dest, err := os.Rea dLi nk( " lin k_f ile.tx t")`

A hard link creates a new pointer to the same place. A file will only be deleted from disk after all links are removed. Hard links only work on the same file system. A hard link is what you might consider a 'normal' link.

A symbolic link, or soft link, only reference other files by name. They can point to files on different filesystems. Not all systems support symlinks.

Read and Write

write bytes to file	`n, err := file.W rit e([ ]by te( " hello, wor "))`
write string to file	`n, err := file.W rit eSt rin g("H ello, world`
write at offset	`n, err := file.W rit eAt ([] byt e("H ell o")`
read to byte	`n, err := file.R ead (by teS lice)`
read exactly n bytes	`n, err := io.Rea dFu ll( file, byteSlice)`
read at least n bytes	`n, err := io.Rea dAt Lea st( file, byteSlice, tes)`

By **cizixs** (cizixs)
cheatography.com/cizixs/
cizixs.com

Published 26th September, 2017.
Last updated 26th September, 2017.

Cheatography

go files Cheat Sheet

by cizixs (cizixs) via cheatography.com/43449/cs/12942/

Read and Write (cont)

read all bytes of a file	`byteSlice, err := ioutil.Re adA ll( fi le)`
read from offset	`n, err := file.R ead At( byt eSlice, 1 0)`

Work with directories

create a directory	`err := os.Mkd ir( " myD ir", 0600)`
recursively create a directory	`err := os.Mkd irA ll( " dir /su bdi r/m yDi -r", 0600)`
delete a directory recursively	`err := os.Rem ove All ("di r/")`
list directory files	`fileInfo, err := ioutil.Re adD ir( ".")`

Shortcuts

quick read from file	`byteSlice, err := ioutil.Re adF ile ("te st.t xt ")`
quick write to file	`err := ioutil.Wr ite Fil e("t est.tx t", []byte ("He llo " ), 0666)`
copy file	`n, err := io.Cop y(n ewFile, origin File)`
write string to file	`io.Wri teS tri ng( file, " Hello, world")`

Temporary files and directories

create temp dir	`ioutil.Te mpD ir(dir, prefix string) (name string, err erro r)`
create temp file	`ioutil.Te mpF ile (dir, prefix string) (f *os.File, err err or)`

References

- Working with Files in Go
- golang `os` standard library
- golang `ioutil` standard library
- golang `iou` standard library

By **cizixs** (cizixs)
cheatography.com/cizixs/
cizixs.com

Published 26th September, 2017.
Last updated 26th September, 2017.