NAME

gg – grep Go-language source code

SYNOPSIS

gg [*options*] *acdiknoprstv regexp* [*file* ...]

DESCRIPTION

gg is grep (g/RE/p) with flag-enabled Go token classes: identifiers, package names, numbers, comments, keywords, and the like. The flags and classes are "acdiknoprstv" in any order and combination.

a: search in All of the following

c: search in Comments (//... or /*...*/)

d: search in Defined non-types (iota, nil, new, true,...)

i: search in Identifiers ([alphabetic][alphabetic | numeric]*)

k: search in Keywords (if, for, func, go, ...)

n: search in Numbers ("255" matches 255, 0.255, 1e255)

o: search in Operators (, + - * / [ ] { } ( ) >> ...)

p: search in Package names

r: search in Rune literals (’a’, ’\U00101234’)

s: search in Strings (quoted or raw)

t: search in Types (bool, int, float64, map, ...)

v: search in Values (255 is 0b11111111, 0377, 255, 0xff)

gg combines lexical analysis and Go-native pattern matching to extend **grep**(1) for Go developers. The search is restricted, seeking matches only in chosen token classes. A search in number literals can match *values*, "v 255" matches the numeric value 255 in source code as 0b1111_1111, 0377, 0o377, 255, 0xff, etc. Go’s robust linear-cost regular expression engine is Unicode aware and supports many Perl extensions, so numbers in identifiers are found by "gg i [0-9]" or "gg i [\d]" and comments containing math symbols are found by "gg c \p{Sm}" (with appropriate shell escaping).

gg searches files named on the command line or in a list of filenames provided the "-list" argument. When neither is provided, gg reads filenames from the standard input. This last is useful in shell pipelines such as "find . -name "*.go" | gg k fallthrough"

Files are Go source files or directories. Source files include typical ".go" files; compressed ".go" files named ".go.bz2", ".go.gz", or ".go.zst" for Bzip2, Gzip, and ZStandard compression formats; archives of any such files in the formats "a.cpio", "a.tar", or "a.zip"; or, finally, compressed archives as in "a.cpio.bz2" and "a.tar.gz". If a named file is a directory then all Go source files in that directory are scanned without visiting subdirectories. With the "-r" flag enabled, named directories are processed recursively, finding and scanning each Go source file in that directory’s hierarchy.

OPTIONS

–cpu=*n*

Set the number of CPUs to use. Default is all.

–go=*bool*

Limit search to ".go" files. Default is true.

–list=*file*

Search files listed one per line in the named file.

–log=*file*

Write a log of execution details to a named file. The special file names "[stdout]" and "[stderr]" refer to the stdout and stderr streams.

–output=*file*

gg output is normally to stdout but may be directed to a named file. The special names "[stdout]" and "[stderr]" refer to the stdout and stderr streams.

–r=*bool*

Search directories recursively. Default is false.

−visible=*bool*

Restrict search to visible files, those with names that do not start with "." (in the shell tradition). Default is true.

acdiknoprstvCDIKNOPRSTV

The Go token class flags have an upper case negative form to disable the indicated class. Used with "a", "aCS" means "search all tokens except comments and strings."

EXAMPLES

To search for comments containing "case" (ignoring switch statements) in the ".go" files of the current working directory, use the command:

```
gg c case .
```

To find number literals containing the digits 42 in Go files anywhere in the current directory's hierarchy, use the command:

```
gg -r n 42 .
```

Find numbers with values of 255 (0b1111_1111, 0377, 0o377, 255, 0xff) in the gzipped **tar**(1) archive omega with the command:

```
gg v 255 omega.tar.gz
```

AUTHOR

Michael T. Jones (https://github.com/MichaelTJones)

SEE ALSO

https://golang.org/pkg/regexp/syntax/
https://en.wikipedia.org/wiki/Unicode_character_property