NAME

ctags − create a tags file

SYNOPSIS

ctags [**−BFadtuwvx**] [**−f** *tagsfile*] *name ...*

DESCRIPTION

ctags makes a tags file for ex(1) from the specified C, Pascal, Fortran, YACC, lex, and lisp sources. A tags file gives the locations of specified objects in a group of files. Each line of the tags file contains the object name, the file in which it is defined, and a search pattern for the object definition, separated by white-space. Using the *tags* file, ex(1) can quickly locate these object definitions. Depending upon the options provided to **ctags**, objects will consist of subroutines, typedefs, defines, structs, enums and unions.

−B use backward searching patterns (?...?).

−F use forward searching patterns (/.../) (the default).

−a append to *tags* file.

−d create tags for #defines that don't take arguments; #defines that take arguments are tagged automatically.

−f Places the tag descriptions in a file called *tagsfile*. The default behaviour is to place them in a file called *tags*.

−t create tags for typedefs, structs, unions, and enums.

−u update the specified files in the *tags* file, that is, all references to them are deleted, and the new values are appended to the file. (Beware: this option is implemented in a way which is rather slow; it is usually faster to simply rebuild the *tags* file.)

−v An index of the form expected by vgrind(1) is produced on the standard output. This listing contains the object name, file name, and page number (assuming 64 line pages). Since the output will be sorted into lexicographic order, it may be desired to run the output through sort(1). Sample use:

```
ctags −v files | sort −f > index
vgrind −x index
```

−w suppress warning diagnostics.

−x **ctags** produces a list of object names, the line number and file name on which each is defined, as well as the text of that line and prints this on the standard output. This is a simple index which can be printed out as an off-line readable function index.

Files whose names end in '.c' or '.h' are assumed to be C source files and are searched for C style routine and macro definitions. Files whose names end in '.y' are assumed to be YACC source files. Files whose names end in '.l' are assumed to be lisp files if their first non-blank character is ';', '(', or '[', otherwise, they are treated as lex files. Other files are first examined to see if they contain any Pascal or Fortran routine definitions, and, if not, are searched for C style definitions.

The tag main is treated specially in C programs. The tag formed is created by prepending *M* to the name of the file, with the trailing '.c' and any leading pathname components removed. This makes use of **ctags** practical in directories with more than one program.

Yacc and lex files each have a special tag. *Yyparse* is the start of the second section of the yacc file, and *yylex* is the start of the second section of the lex file.

FILES

`tags` default output tags file

DIAGNOSTICS

`ctags` exits with a value of 1 if an error occurred, 0 otherwise. Duplicate objects are not considered errors.

SEE ALSO

`ex`(1), `vi`(1)

BUGS

Recognition of *functions*, *subroutines* and *procedures* for FORTRAN and Pascal is done in a very simple-minded way. No attempt is made to deal with block structure; if you have two Pascal procedures in different blocks with the same name you lose. **`ctags`** doesn't understand about Pascal types.

The method of deciding whether to look for C, Pascal or FORTRAN functions is a hack.

`ctags` relies on the input being well formed, and any syntactical errors will completely confuse it. It also finds some legal syntax confusing; for example, since it doesn't understand `#ifdef`'s (incidentally, that's a feature, not a bug), any code with unbalanced braces inside `#ifdef`'s will cause it to become somewhat disoriented. In a similar fashion, multiple line changes within a definition will cause it to enter the last line of the object, rather than the first, as the searching pattern. The last line of multiple line `typedef`'s will similarly be noted.

HISTORY

The **`ctags`** command appeared in 3.0BSD.