## NAME

**xstr -** extract strings from C programs to implement shared strings

## SYNOPSIS

**xstr** [**-cv**] [**-**] [*file* ...]

## DESCRIPTION

The **xstr** utility maintains a file *strings* into which strings in component parts of a large program are hashed. These strings are replaced with references to this common area. This serves to implement shared constant strings, most useful if they are also read-only.

The following options are available:

**-** Read from the standard input.

**-c** Extract the strings from the C source *file* or the standard input (-), replacing string references by expressions of the form (&xstr[number]) for some *number*. An appropriate declaration of *xstr* is prepended to the file. The resulting C text is placed in the file *x.c*, to then be compiled. The strings from this file are placed in the *strings* data base if they are not there already. Repeated strings and strings which are suffixes of existing strings do not cause changes to the data base.

**-v** Verbose mode.

After all components of a large program have been compiled a file *xs.c* declaring the common *xstr* space can be created by a command of the form

```
xstr
```

The file *xs.c* should then be compiled and loaded with the rest of the program. If possible, the array can be made read-only (shared) saving space and swap overhead.

The **xstr** utility can also be used on a single file. A command

```
xstr name
```

creates files *x.c* and *xs.c* as before, without using or affecting any *strings* file in the same directory.

It may be useful to run **xstr** after the C preprocessor if any macro definitions yield strings or if there is conditional code which contains strings which may not, in fact, be needed. An appropriate command sequence for running **xstr** after the C preprocessor is:

```
cc -E name.c | xstr -c -
cc -c x.c
mv x.o name.o
```

The **xstr** utility does not touch the file *strings* unless new items are added, thus make(1) can avoid remaking *xs.o* unless truly necessary.

## FILES

| | |
|---|---|
| *strings* | data base of strings |
| *x.c* | massaged C source |
| *xs.c* | C source for definition of array *xstr* |
| */tmp/xs** | temporary file when "xstr name" does not touch *strings* |

## SEE ALSO

mkstr(1)

## HISTORY

The **xstr** command appeared in 3.0BSD.

## BUGS

If a string is a suffix of another string in the data base, but the shorter string is seen first by **xstr** both strings will be placed in the data base, when just placing the longer one there will do.