NAME

checkup – RCS/SCCS check-in status

SYNOPSIS

checkup *[options] [file-specifications]*

DESCRIPTION

Checkup examines one or more text files, and shows which ones have not been archived with either *rcs* or *sccs*. It also shows those which have been modified since the last check-in.

Checkup scans each filename given as an argument, looking first for a corresponding *rcs* archive, then for an *sccs* archive. For each file, **checkup** determines if the file

- has been archived,
- has been locked for modification,
- has been modified since its check-in date, or has a modification date older than the check-in date.

If a directory name is given, **checkup** scans all files in the directory. In either case, **checkup** attempts to display the offending files in the context of a directory tree. The directory-tree display is written to the standard error stream. **Checkup** writes to the standard output a list of the absolute pathnames for each file. This list may be piped into other utilities, such as the directory editor **ded**.

OPTIONS

–a permits **checkup** to examine directories (and their subdirectories) whose leafnames begin with ".". Also, show binary files. These are otherwise ignored.

–c overrides the convention that filenames written to standard output are those for which **checkup** finds something to report. Instead, the filenames are the complete set of archived files selected by the other options. Each line contains the revision code for the file, followed by its name. This option is used to generate configuration lists from a set of working files.

–d debug option forces **checkup** to show all filenames found. The forced-names are marked "**(ok)**" in the tree listing.

–i *string*
directs **checkup** to ignore all files matching the pattern in *string*. The wildcard characters "*" and "?" are interpreted as in the POSIX shell.

–l *file*
reroutes the directory-tree display (normally written to standard error) to the specified file.

–L causes **checkup** to process symbolic-link targets. Ordinarily these are ignored.

–o directs **checkup** to report obsolete files (i.e., those archives for which no corresponding working-file was found). If both "**–r**" and "**–o**" are specified, the "**–r**" option is interpreted as selecting obsolete-files.

If the "**–o**" option is selected, **checkup** will report also on directories which are found in archive directories. Otherwise, it does not scan the contents of archive directories.

–p directs **checkup** to express the filenames written to standard output as relative pathnames. Otherwise they are written as absolute pathnames (i.e., beginning with "/").

–q makes the listing less verbose (i.e., suppresses display of the directory tree). If standard output is not piped to a file, the list of absolute filenames will be shown on your terminal instead.

–r *REV*
reports all working files whose highest version is below *REV*. For example, "**–r**2" will report all files which are checked in, but having version numbers below "2.0".

A "+" sign may be appended to the "**–r**" option to cause it to reverse the normal order of comparison. For example, "**–r**2+" causes checkup to report files having version numbers above "2.0".

−s same as "**−q**".

−t directs **checkup** to suppress files whose extension ends in a default list: ".bak", ".i", ".log", ".out" and ".tmp".

−v makes the display more verbose; the names of files which cannot be opened are reported.

−x *string*
specifies an extension (filename suffix). All filenames ending in this extension are ignored. The first character of the string doubles as a delimiter (e.g., "."). If it is repeated in the string, **checkup** parses two extensions. The first extension, if any, is used to conditionally ignore the second. That is, if a file with the first extension exists, the file with the second is ignored. In either case, wildcards are permitted in the target extension as in the "**−i**" option.

Multiple instances of the "**−i**" and "**−x**" options may be used. **Checkup** tests files against the exclusion options from right-to-left.

OPERATIONS

An example of the use of **checkup** is shown below:

```
bsd4.2(64) checkup -t -x.e.c ~/traces/lib
** path = //dickey/local/dickey/traces/lib
   1:     //dickey/local/dickey/traces/lib/
   2:      |-- access/
   3:      |-- das/
   4:      |-- das+/
   5:      |-- report/
   6:      |-- traces/
bsd4.2(65) checkup  ~/traces/lib
** path = //dickey/local/dickey/traces/lib
   1:     //dickey/local/dickey/traces/lib/
   2:      |-- access/
   3:      |---|-- lincnt.out (not archived)
   4:      |---|-- lint.out (not archived)
   5:      |-- das/
   6:      |---|-- das.c (not archived)
   7:      |---|-- dbdump.c (not archived)
   8:      |---|-- dbload.c (not archived)
   9:      |---|-- dblook.c (not archived)
  10:     |---|-- lincnt.out (not archived)
  11:     |---|-- lint.out (not archived)
  12:     |-- das+/
  13:     |-- report/
  14:     |---|-- lincnt.out (not archived)
  15:     |---|-- lint.out (not archived)
  16:     |-- traces/
```

ENVIRONMENT

Checkup is a C-language program which runs in a portable POSIX environment. Environment variables include:

RCS_DIR
specifies the directory in which **checkup** will find the ",v" files. If not specified, **checkup** assumes "RCS".

SCCS_DIR
specifies the directory in which **checkup** will find the "s." files. If not specified, **checkup** assumes "SCCS".

FILES

Checkup is a single binary file, "checkup".

ANTICIPATED CHANGES

None.

AUTHORS

Thomas E. Dickey <dickey@invisible-island.net>

SEE ALSO

rlog (1), sact (1).