NAME

link2rcs – RCS directory template

SYNOPSIS

link2rcs *[options] [file-specifications]*

DESCRIPTION

Link2rcs makes (an empty) template of an existing directory tree. All directory names are copied; "RCS" directories are transformed into symbolic links pointing to the original tree.

Link2rcs is used to make a working copy of a source-controlled module which is configured in a tree of directories (e.g., bin, lib, include, src, with dependent subdirectories). By making a template directory (i.e., with the same relative pathnames), you may check-out files and continue to develop the module. By making symbolic links to the original set of *rcs* directories, **link2rcs** facilitates the use of *rcs*: **ci** and **co** assume that the *rcs* files are in a subdirectory named "RCS".

OPTIONS

Link2rcs recognizes the following options:

–a process all directory names. Like the **ls** option "**-a**", this enables the processing of names beginning with ".".

–b*num*
directs **link2rcs** to avoid generating directories which do not correspond to a source-directory with the given *rcs* baseline. Use "**-b**0" to suppress directories which do not correspond to any *rcs* baseline.

–d*path*
specifies the destination directory. You must always specify this if you don't specify the "**-s**" option.

–e*env*
(Apollo only) specifies the name of an environment variable which can be used to substitute for the beginning of the link-text.

–f makes links to files as well as directories.

–F makes hard links to files as well as directories.

–m merge against destination, i.e., don't quit if a target directory or link is found.

–n (no-op) causes **link2rcs** to compute the modifications and to display them (according to the setting of "**-q**" and "**-v**"), but not to modify the target directory tree.

–o overwrite files and links in the destination, replacing them with links, either symbolic or hard, depending on whether you have specified the "**-f**" or "**-F**" option.

–r directs link2rcs to compute relative pathnames for the symbolic links which it creates. Normally the links are made with an absolute pathname (e.g., "**/u1/foo/RCS**").

–q (quiet) undoes "**-v**", eliminates most messages.

–s*path*
specifies the source directory. Normally you need not specify this; it defaults to ".".

–v (verbose) shows shell-command equivalents to the modifications which are made.

OPERATIONS

Normally, you use **link2rcs** when constructing a development directory based on an existing source tree. To copy an entire tree, change your shell's working directory to the source directory (i.e., using the "cd" command) and type

`link2rcs –d` *dst_dir* .

where *dst_dir* is the name of the (existing) directory under which you wish to create the template directory, and the "." specifies the current working directory. If you wish to copy only some of the directories, type their names instead of ".".

Link2rcs makes a list of all directories under the source directory and attempts to create a corresponding template under the destination directory. Each time it finds a directory named "RCS" it makes a symbolic link; otherwise it makes directories to match those in the source tree.

If any of the names which **link2rcs** plans to create already exists, it halts with an error message (unless "**-m**" is specified).

When you have created the template tree, you may use **rcsget** to check-out files into the destination tree.

ENVIRONMENT

Link2rcs is written in C and runs in a POSIX environment.

It uses the **DED_CM_LOOKUP** variable (see ded(1)) to decide whether to create a symbolic link or actual directory for CVS, RCS and SCCS. By default, **linkrcs** creates links for RCS and SCCS directories. But if **DEC_CM_LOOKUP** is set, **link2rcs** creates links only if those features are listed in the environment variable.

FILES

Link2rcs consists of a single binary file, **link2rcs**.

ANTICIPATED CHANGES

None.

AUTHORS

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

SEE ALSO

ded, baseline, rcsget, checkout, permit, rcs(1)