NAME

git-mv − Move or rename a file, a directory, or a symlink

SYNOPSIS

git mv <options>... <args>...

DESCRIPTION

Move or rename a file, directory or symlink.

git mv [−v] [−f] [−n] [−k] <source> <destination>
git mv [−v] [−f] [−n] [−k] <source> ... <destination directory>

In the first form, it renames <source>, which must exist and be either a file, symlink or directory, to <destination>. In the second form, the last argument has to be an existing directory; the given sources will be moved into this directory.

The index is updated after successful completion, but the change must still be committed.

OPTIONS

−f, −−force
: Force renaming or moving of a file even if the target exists

−k
: Skip move or rename actions which would lead to an error condition. An error happens when a source is neither existing nor controlled by Git, or when it would overwrite an existing file unless **−f** is given.

−n, −−dry−run
: Do nothing; only show what would happen

−v, −−verbose
: Report the names of files as they are moved.

SUBMODULES

Moving a submodule using a gitfile (which means they were cloned with a Git version 1.7.8 or newer) will update the gitfile and core.worktree setting to make the submodule work in the new location. It also will attempt to update the submodule.<name>.path setting in the **gitmodules(5)** file and stage that file (unless −n is used).

BUGS

Each time a superproject update moves a populated submodule (e.g. when switching between commits before and after the move) a stale submodule checkout will remain in the old location and an empty directory will appear in the new location. To populate the submodule again in the new location the user will have to run "git submodule update" afterwards. Removing the old directory is only safe when it uses a gitfile, as otherwise the history of the submodule will be deleted too. Both steps will be obsolete when recursive submodule update has been implemented.

GIT

Part of the **git(1)** suite