NAME

git−qguard − manage assignment of guards to patches

SYNOPSIS

git qguard [**−l** | **−−list**]
git qguard [**−n** | **−−none**] [<*patch*>] [**−−** [**+**<*guard*> | **−**<*guard*>] **...**]

DESCRIPTION

The **git qguard** command may be used to set, or to clear, the guard assignments for any one patch, or to display the current guard assignments for any one, or for all patches in a **Git−MQ** patch series.

The first form of **git qguard**, characterized by the specification of the **−l** (or **−−list**) option, is used to display a list of all patches in the series, together with the guards, if any, which have been assigned to each; when using this form of the command, it is an error to specify the **−n** (or **−−none**) option, or any other arguments.

The second form of the command is used to set, clear, or display the guard assignments for a single patch, nominated by the <*patch*> argument; if this is unspecified, the topmost currently-applied patch is implied.

In the second form, if the **−n** (or **−−none**) option is specified, all guards, if any, which had been assigned previously, to the nominated patch, are cleared; then, any guards specified by **+**<*guard*> or **−**<*guard*> arguments are assigned.

If **−n** (or **−−none**) is not specified, and no **+**<*guard*> or **−**<*guard*> arguments are specified, then any guards which are currently assigned to the nominated patch are displayed, and remain unchanged.

Note that, when any **+**<*guard*> or **−**<*guard*> arguments are specified, the specified guards will be assigned as *replacements* for any which were previously assigned to the nominated patch; thus, it is unnecessary, (but not forbidden), to specify **−n** (or **−−none**) in conjunction with new guard assignments.

Also note that the **−−** separator is mandatory, if any guards of the **−**<*guard*> form are to be assigned.

Guards control the behaviour of the **git qpush** command, when pushing one or more patches, according to the following strategy:—

- A patch with no guards assigned may be pushed unconditionally;
- otherwise, a patch with negative guards assigned may be pushed if, and only if, *none* of those matches any guard which has been activated, (and not subsequently deactivated), by the **git qselect** command;
- otherwise, a patch with any positive guards assigned may be pushed if, and only if, *every one* of those is matched by a guard which has been activated, (and not subsequently deactivated), by the **git qselect** command.
- The ultimate push destination *must* represent a patch which may be pushed, having satisfied the preceding conditions, in the specified order; otherwise, the **git qpush** command fails, without pushing any patch.
- Provided the patch, specified as the ultimate push destination, may be pushed, any other patches in the path to this destination will be pushed if, and only if, they fully satisfy the preceding sequence of conditions; any which do not will be skipped.

It follows, from the foregoing, that any patch which has the same guard assigned in both the positive, and in the negative sense, can *never* be pushed.

OPTIONS

−l, **−−list**
: Enumerate all registered patches, in series order, and identify the set of guards, if any, which is associated with each; this may not be used in conjunction with the **−−none** option, or with any arguments.

−n, **−−none**
: Clear all guard assignments for the named <*patch*>, (or for the topmost applied patch, if no <*patch*> argument is specified); this is redundant, and silently ignored, if used in conjunction with any **+**<*guard*> or **−**<*guard*> assignment arguments.

EXIT STATUS

On successful completion, **git qguard** reports an exit status code of 0. Any non-zero exit status code indicates that an error occurred.

COMPARISON WITH MERCURIAL QUEUES

Inspired by, and for the most part based on behavioural observation of **Mercurial**'s **MQ** extension, the **git qguard** command exhibits fundamentally the same behaviour as its **hg qguard** counterpart.

AUTHOR

SEE ALSO

git−qpush(1), **git−qselect**(1)