NAME

git−qselect – manage the active guard state of a patch queue

SYNOPSIS

git qselect [**−s** | **−−series**] [**−v** | **−−verbose**]
git qselect [**−n** | **−−none**] [**−−pop**] [**−−reapply**] [<*guard*> **...**]

DESCRIPTION

The **git qselect** command is used to manage the active guard state of a patch queue, and so, in conjunction with the **git qguard** command, control the selection of patches which may be pushed.

Depending on the combination of options and arguments specified, the **git qselect** command supports three distinct modes of operation, namely, in order of precedence from highest to lowest:—

1. Invoked when the **−n** (or **−−none**) option, or any set of one or more <*guard*> arguments is specified, the **−n** (or **−−none**) option, if specified, initially ensures that any currently active guards are deactivated; the active guard state is then set to *exactly* match any particular set of <*guard*> arguments which are specified.

   Note that, in this highest precedence mode of operation, the effect of <*guard*> arguments is *not* cumulative; thus, specification of the **−n** (or **−−none**) option together with any non-empty set of <*guard*> arguments, while permitted, is effectively redundant. However, the **−n** (or **−−none**) option *is* required, when the intent is to *clear* currently-active guards, and leave the queue with all guards deactivated; in this case, the **−n** (or **−−none**) option *must* be specified, with no accompanying set of <*guard*> arguments.

2. Invoked only when the **−s** (or **−−series**) option is specifed, and neither the **−n** (or **−−none**) option, nor any <*guard*> argument is present, the command will display an alpha-numerically sorted list of guards which have been defined by the **git qguard** command, and recorded within the series file, regardless of the active guard state of the patch queue.

   The display format, in this mode of operation, may be agumented to include a count of patches to which each guard is assigned, (and also a count of patches to which no guards have been assigned, designated by the reserved pseudo-guard name “**NONE**”), by specifying the **−v** (or **−−verbose**) option, in conjunction with **−s** (or **−−series**).

3. Invoked when no <*guard*> arguments are, and neither the **−n** (or **−−none**) option, nor the **−s** (or **−−series**) option is specified, this lowest precedence mode of operation simply produces a display of the currently active guards, if any, or reports that no guards are active, otherwise.

Note that it is possible for the **git qselect**, and **git qguard** commands to modify the guard state of applied patches, without changing the applied state of the patch series; invoking **git qselect** with the **−−pop**, or **−−reapply** options modifies this behaviour.

OPTIONS

−n, **−−none**
: Selects the highest precedence mode of operation, and causes deactivation of any currently active guards.

−−pop
: In the case where guarded patches, (in the prevailing guard state after specified changes, if any, have been instated), precede the topmost applied patch, pop patches from the stack until no such patches remain within the applied series.

−−reapply
: Note the identity of the topmost applied patch, and proceed as for the **−−pop** option; then, provided the former topmost patch remains unguarded, in the currently active guard state, push patches, skipping any which *are* guarded, until the formerly identified patch again becomes topmost.

−s, **−−series**
: Ignored if any <*guard*> argument, or the **−−none** option is also specified; otherwise selects the intermediate precedence mode of operation, to display a list of all guards which have been defined in

the series file.

−v, **−−verbose**
In the case where the effect of the **−−series** option is dominant, include patch counts for each defined guard, and the tally of unguarded patches, in the displayed list.

EXIT STATUS

On successful completion, **git qselect** 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 qselect** command exhibits fundamentally the same behaviour as its **hg qselect** counterpart.

CAVEATS AND BUGS

It may be observed that certain combinations of options, and <*guard*> arguments make no sense. Allowing execution to continue, when any such nonsensical combination is specified, may be considered a bug. However, this is not caught; rather, execution *does* continue, on the basis of selection of the highest precedence applicable operating mode, and inconsistently specified options are silently ignored.

AUTHOR

SEE ALSO

git−qguard(1), **git−qpop**(1), **git−qpush**(1), **git−qseries**(1)