NAME

git-send-pack - Push objects over Git protocol to another repository

SYNOPSIS

git send-pack [--all] [--dry-run] [--force] [--receive-pack=<git-receive-pack>] [--verbose] [--thin] [<host>:]<directory

DESCRIPTION

Usually you would want to use *git push*, which is a higher-level wrapper of this command, instead. See **git-push(1)**.

Invokes *git-receive-pack* on a possibly remote repository, and updates it from the current repository, sending named refs.

OPTIONS

--receive-pack=<git-receive-pack>
: Path to the *git-receive-pack* program on the remote end. Sometimes useful when pushing to a remote repository over ssh, and you do not have the program in a directory on the default $PATH.

--exec=<git-receive-pack>
: Same as --receive-pack=<git-receive-pack>.

--all
: Instead of explicitly specifying which refs to update, update all heads that locally exist.

--stdin
: Take the list of refs from stdin, one per line. If there are refs specified on the command line in addition to this option, then the refs from stdin are processed after those on the command line.

  If *--stateless-rpc* is specified together with this option then the list of refs must be in packet format (pkt-line). Each ref must be in a separate packet, and the list must end with a flush packet.

--dry-run
: Do everything except actually send the updates.

--force
: Usually, the command refuses to update a remote ref that is not an ancestor of the local ref used to overwrite it. This flag disables the check. What this means is that the remote repository can lose commits; use it with care.

--verbose
: Run verbosely.

--thin
: Send a thin pack, which records objects in deltified form based on objects not included in the pack to reduce network traffic.

<host>
: A remote host to house the repository. When this part is specified, *git-receive-pack* is invoked via ssh.

<directory>
: The repository to update.

<ref>...
: The remote refs to update.

SPECIFYING THE REFS

There are three ways to specify which refs to update on the remote end.

With *--all* flag, all refs that exist locally are transferred to the remote side. You cannot specify any *<ref>* if you use this flag.

Without *--all* and without any <*ref*>, the heads that exist both on the local side and on the remote side are updated.

When one or more <*ref*> are specified explicitly (whether on the command line or via --stdin), it can be either a single pattern, or a pair of such pattern separated by a colon : (this means that a ref name cannot have a colon in it). A single pattern <*name*> is just a shorthand for <*name*>:<*name*>.

Each pattern pair consists of the source side (before the colon) and the destination side (after the colon). The ref to be pushed is determined by finding a match that matches the source side, and where it is pushed is determined by using the destination side. The rules used to match a ref are the same rules used by *git rev-parse* to resolve a symbolic ref name. See **git-rev-parse(1)**.

- It is an error if <src> does not match exactly one of the local refs.
- It is an error if <dst> matches more than one remote refs.
- If <dst> does not match any remote ref, either
  - it has to start with refs/; <dst> is used as the destination literally in this case.
  - <src> == <dst> and the ref that matched the <src> must not exist in the set of remote refs; the ref matched <src> locally is used as the name of the destination.

Without *--force*, the <src> ref is stored at the remote only if <dst> does not exist, or <dst> is a proper subset (i.e. an ancestor) of <src>. This check, known as fast-forward check, is performed in order to avoid accidentally overwriting the remote ref and lose other peoples commits from there.

With *--force*, the fast-forward check is disabled for all refs.

Optionally, a <ref> parameter can be prefixed with a plus + sign to disable the fast-forward check only on that ref.

GIT

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