NAME

git-receive-pack - Receive what is pushed into the repository

SYNOPSIS

git-receive-pack <directory>

DESCRIPTION

Invoked by *git send-pack* and updates the repository with the information fed from the remote end.

This command is usually not invoked directly by the end user. The UI for the protocol is on the *git send-pack* side, and the program pair is meant to be used to push updates to remote repository. For pull operations, see **git-fetch-pack(1)**.

The command allows for creation and fast-forwarding of sha1 refs (heads/tags) on the remote end (strictly speaking, it is the local end *git-receive-pack* runs, but to the user who is sitting at the send-pack end, it is updating the remote. Confused?)

There are other real-world examples of using update and post-update hooks found in the Documentation/howto directory.

git-receive-pack honours the receive.denyNonFastForwards config option, which tells it if updates to a ref should be denied if they are not fast-forwards.

OPTIONS

<directory>
: The repository to sync into.

PRE-RECEIVE HOOK

Before any ref is updated, if $GIT_DIR/hooks/pre-receive file exists and is executable, it will be invoked once with no parameters. The standard input of the hook will be one line per ref to be updated:

sha1-old SP sha1-new SP refname LF

The refname value is relative to $GIT_DIR; e.g. for the master head this is refs/heads/master. The two sha1 values before each refname are the object names for the refname before and after the update. Refs to be created will have sha1-old equal to 0{40}, while refs to be deleted will have sha1-new equal to 0{40}, otherwise sha1-old and sha1-new should be valid objects in the repository.

This hook is called before any refname is updated and before any fast-forward checks are performed.

If the pre-receive hook exits with a non-zero exit status no updates will be performed, and the update, post-receive and post-update hooks will not be invoked either. This can be useful to quickly bail out if the update is not to be supported.

UPDATE HOOK

Before each ref is updated, if $GIT_DIR/hooks/update file exists and is executable, it is invoked once per ref, with three parameters:

$GIT_DIR/hooks/update refname sha1-old sha1-new

The refname parameter is relative to $GIT_DIR; e.g. for the master head this is refs/heads/master. The two sha1 arguments are the object names for the refname before and after the update. Note that the hook is called before the refname is updated, so either sha1-old is 0{40} (meaning there is no such ref yet), or it should match what is recorded in refname.

The hook should exit with non-zero status if it wants to disallow updating the named ref. Otherwise it should exit with zero.

Successful execution (a zero exit status) of this hook does not ensure the ref will actually be

updated, it is only a prerequisite. As such it is not a good idea to send notices (e.g. email) from this hook. Consider using the post-receive hook instead.

POST-RECEIVE HOOK

After all refs were updated (or attempted to be updated), if any ref update was successful, and if $GIT_DIR/hooks/post-receive file exists and is executable, it will be invoked once with no parameters. The standard input of the hook will be one line for each successfully updated ref:

sha1-old SP sha1-new SP refname LF

The refname value is relative to $GIT_DIR; e.g. for the master head this is refs/heads/master. The two sha1 values before each refname are the object names for the refname before and after the update. Refs that were created will have sha1-old equal to 0{40}, while refs that were deleted will have sha1-new equal to 0{40}, otherwise sha1-old and sha1-new should be valid objects in the repository.

Using this hook, it is easy to generate mails describing the updates to the repository. This example script sends one mail message per ref listing the commits pushed to the repository:

```
#!/bin/sh
# mail out commit update information.
while read oval nval ref
do
if expr $oval : 0*$ >/dev/null
then
echo Created a new ref, with the following commits:
git rev-list --pretty $nval
else
echo New commits:
git rev-list --pretty $nval ^$oval
fi |
mail -s Changes to ref $ref commit-list@mydomain
done
exit 0
```

The exit code from this hook invocation is ignored, however a non-zero exit code will generate an error message.

Note that it is possible for refname to not have sha1-new when this hook runs. This can easily occur if another user modifies the ref after it was updated by *git-receive-pack*, but before the hook was able to evaluate it. It is recommended that hooks rely on sha1-new rather than the current value of refname.

POST-UPDATE HOOK

After all other processing, if at least one ref was updated, and if $GIT_DIR/hooks/post-update file exists and is executable, then post-update will be called with the list of refs that have been updated. This can be used to implement any repository wide cleanup tasks.

The exit code from this hook invocation is ignored; the only thing left for *git-receive-pack* to do at that point is to exit itself anyway.

This hook can be used, for example, to run git update-server-info if the repository is packed and is served via a dumb transport.

```
#!/bin/sh
exec git update-server-info
```

SEE ALSO

git-send-pack(1), **gitnamespaces(7)**

GIT

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