

**NAME**

git-remote-ext – Bridge smart transport to external command.

**SYNOPSIS**

```
git remote add <nick> "ext::<command>[ <arguments>...]"
```

**DESCRIPTION**

This remote helper uses the specified *<command>* to connect to a remote Git server.

Data written to stdin of the specified *<command>* is assumed to be sent to a `git://` server, `git-upload-pack`, `git-receive-pack` or `git-upload-archive` (depending on situation), and data read from stdout of *<command>* is assumed to be received from the same service.

Command and arguments are separated by an unescaped space.

The following sequences have a special meaning:

'% '

Literal space in command or argument.

%%

Literal percent sign.

%s

Replaced with name (`receive-pack`, `upload-pack`, or `upload-archive`) of the service Git wants to invoke.

%S

Replaced with long name (`git-receive-pack`, `git-upload-pack`, or `git-upload-archive`) of the service Git wants to invoke.

%G (must be the first characters in an argument)

This argument will not be passed to *<command>*. Instead, it will cause the helper to start by sending `git://` service requests to the remote side with the service field set to an appropriate value and the repository field set to rest of the argument. Default is not to send such a request.

This is useful if remote side is `git://` server accessed over some tunnel.

%V (must be first characters in argument)

This argument will not be passed to *<command>*. Instead it sets the `vhost` field in the `git://` service request (to rest of the argument). Default is not to send `vhost` in such request (if sent).

**ENVIRONMENT VARIABLES**

GIT\_TRANSLOOP\_DEBUG

If set, prints debugging information about various reads/writes.

**ENVIRONMENT VARIABLES PASSED TO COMMAND**

GIT\_EXT\_SERVICE

Set to long name (`git-upload-pack`, etc...) of service helper needs to invoke.

GIT\_EXT\_SERVICE\_NOPREFIX

Set to long name (`upload-pack`, etc...) of service helper needs to invoke.

**EXAMPLES**

This remote helper is transparently used by Git when you use commands such as `"git fetch <URL>"`, `"git clone <URL>"`, `"git push <URL>"` or `"git remote add <nick> <URL>"`, where *<URL>* begins with **ext::**. Examples:

```
"ext::ssh -i /home/foo/.ssh/somekey user@host.example %S foo/repo"
```

Like `host.example:foo/repo`, but use `/home/foo/.ssh/somekey` as `keypair` and `user` as `user` on remote side. This avoids needing to edit `.ssh/config`.

"ext::socat -t3600 - ABSTRACT-CONNECT:/git-server %G/somerepo"

Represents repository with path /somerepo accessible over git protocol at abstract namespace address /git-server.

"ext::git-server-alias foo %G/repo"

Represents a repository with path /repo accessed using the helper program "git-server-alias foo". The path to the repository and type of request are not passed on the command line but as part of the protocol stream, as usual with git:// protocol.

"ext::git-server-alias foo %G/repo %Vfoo"

Represents a repository with path /repo accessed using the helper program "git-server-alias foo". The hostname for the remote server passed in the protocol stream will be "foo" (this allows multiple virtual Git servers to share a link-level address).

"ext::git-server-alias foo %G/repo% with% spaces %Vfoo"

Represents a repository with path **/repo with spaces** accessed using the helper program "git-server-alias foo". The hostname for the remote server passed in the protocol stream will be "foo" (this allows multiple virtual Git servers to share a link-level address).

"ext::git-ssl foo.example /bar"

Represents a repository accessed using the helper program "git-ssl foo.example /bar". The type of request can be determined by the helper using environment variables (see above).

## SEE ALSO

**gitremote-helpers(7)**

## GIT

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