



Rocky Enterprise Linux 9.2 Manual Pages on command 'git-name-rev.1'

C:\>man git-name-rev.1

GIT-NAME-REV(1) Git Manual GIT-NAME-REV(1)

NAME

git-name-rev - Find symbolic names for given revs

SYNOPSIS

```
git name-rev [--tags] [--refs=<pattern>]
              ( --all | --stdin | <commit-ish>... )
```

DESCRIPTION

Finds symbolic names suitable for human digestion for revisions given in any format parsable by git rev-parse.

OPTIONS

--tags

Do not use branch names, but only tags to name the commits

--refs=<pattern>

Only use refs whose names match a given shell pattern. The pattern can be one of branch name, tag name or fully qualified ref name. If given multiple times, use refs whose names match any of the given shell patterns. Use --no-refs to clear any previous ref patterns given.

--exclude=<pattern>

Do not use any ref whose name matches a given shell pattern. The pattern can be one of branch name, tag name or fully qualified ref name. If given multiple times, a ref will be excluded when it matches any of the given patterns. When used together with --refs, a ref will be used as a match only when it matches

at least one --refs pattern and does not match any --exclude patterns. Use --no-exclude to clear the list of exclude patterns.

--all

List all commits reachable from all refs

--stdin

Transform stdin by substituting all the 40-character SHA-1 hexes (say \$hex) with "\$hex (\$rev_name)". When used with --name-only, substitute with "\$rev_name", omitting \$hex altogether. Intended for the scripter's use.

--name-only

Instead of printing both the SHA-1 and the name, print only the name. If given with --tags the usual tag prefix of "tags/" is also omitted from the name, matching the output of git-describe more closely.

--no-undefined

Die with error code != 0 when a reference is undefined, instead of printing undefined.

--always

Show uniquely abbreviated commit object as fallback.

EXAMPLES

Given a commit, find out where it is relative to the local refs. Say somebody wrote you about that fantastic commit 33db5f4d9027a10e477ccf054b2c1ab94f74c85a. Of course, you look into the commit, but that only tells you what happened, but not the context.

Enter git name-rev:

```
% git name-rev 33db5f4d9027a10e477ccf054b2c1ab94f74c85a
33db5f4d9027a10e477ccf054b2c1ab94f74c85a tags/v0.99~940
```

Now you are wiser, because you know that it happened 940 revisions before v0.99.

Another nice thing you can do is:

```
% git log | git name-rev --stdin
```

GIT

Part of the git(1) suite

Git 2.25.1

02/08/2023

GIT-NAME-REV(1)