**选项**

**-v**

**--版本**

打印 git 程序源自的 **Git** 套件版本。

此选项在内部转换为并接受与 [git-version[1]](https://git-scm.com/docs/git-version) 命令相同的选项。如果还给出了，则它优先于 。git version ...--help--version

**-h**

**--帮助**

打印概要和最常用命令的列表。如果给出了选项 或，则打印所有可用的命令。如果 Git 命令被命名为此选项，则会显示该命令的手册页。--all-a

其他选项可用于控制手册页的显示方式。有关详细信息[，请参阅 git-help[1]](https://git-scm.com/docs/git-help)，因为在内部已转换为 .git --help ...git help ...

**-C <路径>**

运行，就好像 git 是在**<path>**而不是当前工作目录中启动的一样。当给出多个选项时，将相对于前面的 解释每个后续非绝对选项。如果**<path>**存在但为空，例如，则当前工作目录保持不变。-C-C <path>-C <path>-C ""

此选项会影响需要类似路径名的选项，并且它们对路径名的解释将相对于该选项引起的工作目录进行。例如，以下调用是等效的：--git-dir--work-tree-C

git --git-dir=a.git --work-tree=b -C c status

git --git-dir=c/a.git --work-tree=c/b status

**-c <name>=<值>**

将配置参数传递给命令。给定的值将覆盖配置文件中的值。<name>应采用与 **git 配置**列出的格式相同的格式（由点分隔的子项）。

请注意，允许省略 in 并设置为布尔 true 值（就像在配置文件中一样）。包括等于但具有空值（如）将设置为空字符串，该空字符串将转换为 。=git -c foo.bar ...foo.bar[foo]bargit -c foo.bar= ...foo.bargit config --type=boolfalse

**--config-env=<name>=<envvar>**

例如，为配置变量**<name>**一个值，其中 <envvar> 是要从中检索值的环境变量的名称。与没有直接将值设置为空字符串的快捷方式不同，环境变量本身必须设置为空字符串。如果环境中不存在 ，则为错误。 可能不包含等号，以避免与包含等号的歧义。-c <name>=<value>-c<envvar><envvar><name>

这对于以下情况很有用：您希望将临时配置选项传递给 git，但在操作系统上这样做，其中其他进程可能能够读取您的 cmdline（例如 ），但不能读取您的环境（例如 ）。该行为是 Linux 上的默认设置，但可能不适用于您的系统。/proc/self/cmdline/proc/self/environ

请注意，这可能会增加变量的安全性，例如敏感信息是值的一部分，但不会增加例如 其中，敏感信息可以是密钥的一部分。http.extraHeaderurl.<base>.insteadOf

**--exec-path[=<path>]**

核心 Git 程序安装位置的路径。这也可以通过设置GIT\_EXEC\_PATH环境变量来控制。如果未指定路径，**git** 将打印当前设置，然后退出。

**--html-path**

打印路径，不带尾部斜杠，Git 的 HTML 文档将安装并退出。

**--人路**

打印此版本 Git 的手册页的手册页的 manpath （请参见 ），然后退出。man(1)

**--信息路径**

打印安装并退出记录此版本 Git 的 Info 文件的路径。

**-p**

**--分页**

如果标准输出是终端，则将所有输出管道连接到**较少**的（或者如果设置，则$PAGER）。这将覆盖配置选项（请参阅下面的“配置机制”部分）。pager.<cmd>

**-P**

**--无寻呼机**

不要将 Git 输出通过管道传送到寻呼机中。

**--git-dir=<path>**

设置存储库的路径（“.git”目录）。这也可以通过设置环境变量来控制。它可以是当前工作目录的绝对路径或相对路径。GIT\_DIR

使用此选项（或环境变量）指定“.git”目录的位置会关闭尝试查找具有“.git”子目录的目录的存储库发现（这是发现存储库和工作树的顶层的方式），并告诉 Git 您处于工作树的顶层。如果您不在工作树的顶级目录中，则应使用选项（或环境变量）告诉 Git 工作树的顶级位置GIT\_DIR--work-tree=<path>GIT\_WORK\_TREE

如果你只是想运行 git，就好像它是在 中启动的一样，那么使用 .<path>git -C <path>

**--工作树=<路径>**

设置工作树的路径。它可以是绝对路径，也可以是相对于当前工作目录的路径。这也可以通过设置GIT\_WORK\_TREE环境变量和core.worktree配置变量来控制（有关更详细的讨论，请参阅[git-config[1]](https://git-scm.com/docs/git-config)中的core.worktree）。

**--命名空间 = <路径>**

设置 Git 命名空间。参见 [gitnamespaces[7]](https://git-scm.com/docs/gitnamespaces) 了解更多详情。等效于设置环境变量。GIT\_NAMESPACE

**--超级前缀=<路径>**

目前仅供内部使用。设置一个前缀，该前缀提供从存储库上方到其根目录的路径。一种用途是为子模块提供有关调用它的超级项目的上下文。

**--裸露**

将存储库视为裸存储库。如果未设置GIT\_DIR环境，则将其设置为当前工作目录。

**--无替换对象**

不要使用替换引用来替换 Git 对象。参见 [git-replace[1]](https://git-scm.com/docs/git-replace) 了解更多信息。

**--文字路径规范**

从字面上处理路径规范（即没有通球，没有路径规范魔法）。这等效于将环境变量设置为 。GIT\_LITERAL\_PATHSPECS1

**--glob-pathspecs**

将“glob”魔法添加到所有路径中。这等效于将环境变量设置为 。

禁用单个路径上的通配可以使用 pathpec 魔术“:(literal）”来完成GIT\_GLOB\_PATHSPECS1

**--noglob-pathspecs**

将“字面”魔术添加到所有路径中。这等效于将环境变量设置为 。

在单个路径上启用通配可以使用路径规范魔术“:(glob）”来完成GIT\_NOGLOB\_PATHSPECS1

**--icase-pathspecs**

将“icase”魔术添加到所有路径规范中。这等效于将环境变量设置为 。GIT\_ICASE\_PATHSPECS1

**--无可选锁**

不要执行需要锁的可选操作。这等效于将 设置为 。GIT\_OPTIONAL\_LOCKS0

**--list-cmds=group[，group...]**

按组列出命令。

这是一个内部/实验性选项，将来可能会更改或删除。

支持的组有：内置的，parseopt（使用解析选项的内置命令），main（libexec目录中的所有命令），

其他（所有其他具有git-前缀的命令），list-<category>（参见command-list中的类别.txt），

nohelpers（排除帮助程序命令），别名和config（从config变量完成中检索命令列表。命令）$PATH