# Git

1. 下载安装git

<https://git-scm.com>

1. 运行以下命令使用git（IOS使用终端，windows使用GitBash）

|  |
| --- |
| # 进入需要建立仓库的目录  $ **cd /F/WorkSpace/geekbang/**  snowc@SNOW-CITY MINGW64 /F/WorkSpace/geekbang  $ **mkdir learn\_git**  # 在指定目录建立 learn\_git 仓库项目目录  snowc@SNOW-CITY MINGW64 /F/WorkSpace/geekbang  $ **cd learn\_git/**  # 进入新建的目录  snowc@SNOW-CITY MINGW64 /F/WorkSpace/geekbang/learn\_git  $ ls -al  total 0  drwxr-xr-x 1 snowc 197609 0 6月 27 00:16 ./  drwxr-xr-x 1 snowc 197609 0 6月 27 00:16 ../  # 查询目录 显示目录中无文件  snowc@SNOW-CITY MINGW64 /F/WorkSpace/geekbang/learn\_git  $ **git init**  Initialized empty Git repository in F:/WorkSpace/geekbang/learn\_git/.git/  # 初始化仓库 这时提示 初始化了一个空的git仓库在指定位置  snowc@SNOW-CITY MINGW64 /F/WorkSpace/geekbang/learn\_git (master)  $ ls -al  total 4  drwxr-xr-x 1 snowc 197609 0 6月 27 00:17 ./  drwxr-xr-x 1 snowc 197609 0 6月 27 00:16 ../  drwxr-xr-x 1 snowc 197609 0 6月 27 00:17 .git/  # 查询目录 目录中生成了 .git 目录  snowc@SNOW-CITY MINGW64 /F/WorkSpace/geekbang/learn\_git (master)  $ **git config --global user.name "XXX"**  # 配置git的user.name  snowc@SNOW-CITY MINGW64 /F/WorkSpace/geekbang/learn\_git (master)  $ **git config --global user.email "XXX@XXX"**  # 配置git的user.email  # 配置信息让git记录是谁做了什么操作  snowc@SNOW-CITY MINGW64 /F/WorkSpace/geekbang/learn\_git (master)  $ **git config --global --list**  user.name= XXX  user.email=XXX@XXX  # --list命令查询配置信息  #########################################################################  snowc@SNOW-CITY MINGW64 /F/WorkSpace/geekbang/learn\_git (master)  # 显示所有配置  $ **git config --list**  diff.astextplain.textconv=astextplain  filter.lfs.clean=git-lfs clean -- %f  filter.lfs.smudge=git-lfs smudge -- %f  filter.lfs.process=git-lfs filter-process  filter.lfs.required=true  http.sslbackend=openssl  http.sslcainfo=d:/Program Files/Git/mingw64/ssl/certs/ca-bundle.crt  core.autocrlf=true  core.fscache=true  core.symlinks=false  pull.rebase=false  credential.helper=manager  # 以上为 **git config --system -l 查看系统配置 系统级别 –list简写为-l**  user.name=snowcitygame  user.email=snowcitygame@qq.com  # 以上为 **git config --global-l 查看用户配置 用户级别**  core.repositoryformatversion=0  core.filemode=false  core.bare=false  core.logallrefupdates=true  core.symlinks=false  core.ignorecase=true  # 以上为 **git config –local -l 查看仓库配置 仓库级别** |
| [git config配置](https://www.cnblogs.com/fireporsche/p/9359130.html)  在git中，我们使用git config 命令用来配置git的配置文件，git配置级别主要有以下3类：  1、仓库级别 local 【优先级最高】  2、用户级别 global【优先级次之】  3、系统级别 system【优先级最低】  **git config 常用配置选项:**  git config -e 编辑配置文件  git config --local -e 编辑仓库级别配置文件  git config --global -e 编辑用户级别配置文件  git config --system -e 编辑系统级别配置文件  git config 添加配置项目  git config --global user.email “you@example.com”  git config --global user.name “Your Name”  上面的操作表示添加用户级别的配置信息，也就是说修改用户宿主目录下面的配置文件  **配置文件如何生效的**  对于git来说，配置文件的权重是仓库>全局>系统。Git会使用这一系列的配置文件来存储你定义的偏好，它首先会查找/etc/gitconfig文件（系统级），该文件含有对系统上所有用户及他们所拥有的仓库都生效的配置值。接下来Git会查找每个用户的~/.gitconfig文件（全局级）。最后Git会查找由用户定义的各个库中Git目录下的配置文件.git/config（仓库级），该文件中的值只对当前所属仓库有效。  **增加配置项**  参数 --add  格式: git config [--local|--global|--system] --add section.key value(默认是添加在local配置中)  注意add后面的section.key,value一项都不能少，否则添加失败。  比如我们执行：git config –add cat.name tom #不指定权重 默认为local  **获取一个配置项**  有时候，我们并不需要查看所有配置的值，而是查看某个配置项的值.  命令参数 --get  格式：git config [--local|--global|--system] --get section.key(默认是获取local配置中内容)  我们先往global配置中写入一个cat.name=Tomcat的配置项，再使用git config --get cat.name  结果就是local中的cat.name=Tom，因此git config --get section.key 等价于  git config --local --get section.key  如果获取一个section不存在的key值，不会返回任何值  如果获取一个不存在的section的key值，则会报错  **删除一个配置项**  命令参数 –unset  格式：git config [--local|--global|--system] --unset section.key  删除local配置中的cat.name  git config -–unset cat.name |

1. git操作流程

|  |
| --- |
| snowc@SNOW-CITY MINGW64 /F/WorkSpace/geekbang/learn\_git (master)  $ **git status**  On branch master  No commits yet  nothing to commit (create/copy files and use "git add" to track)  # 查看git仓库状态 没有东西需要提交（新建、复制文件，并使用git add来跟踪这些文件）  snowc@SNOW-CITY MINGW64 /F/WorkSpace/geekbang/learn\_git (master)  $ git status  On branch master  No commits yet  Untracked files:#未跟踪的文件  (use "git add <file>..." to include in what will be committed)  test\_git.txt  nothing added to commit but untracked files present (use "git add" to track)  # 新建或复制一个文件后，提示使用git add来跟踪这些文件  snowc@SNOW-CITY MINGW64 /F/WorkSpace/geekbang/learn\_git (master)  $ **git add test\_git.txt**  或$ git add .  # git add 跟踪指定文件（用.表示跟踪所有未跟踪的文件）  snowc@SNOW-CITY MINGW64 /F/WorkSpace/geekbang/learn\_git (master)  $ git status  On branch master  No commits yet  Changes to be committed:  (use "git rm --cached <file>..." to unstage)#提示取消暂存的命令  new file: test\_git.txt  # 提示进行提交操作  snowc@SNOW-CITY MINGW64 /F/WorkSpace/geekbang/learn\_git (master)  $ **git commit -m "new empty file test\_git.txt"**  [master (root-commit) ccac291] new empty file test\_git.txt  1 file changed, 0 insertions(+), 0 deletions(-)  create mode 100644 test\_git.txt  # git commit -m "注释" 提交操作  snowc@SNOW-CITY MINGW64 /f/WorkSpace/geekbang/learn\_git (master)  $ git status  On branch master  nothing to commit, working tree clean  #提交后提示 没有东西要提交，工作树（工作区）没东西  snowc@SNOW-CITY MINGW64 /f/WorkSpace/geekbang/learn\_git (master)  $ **git status**  On branch master  Changes not staged for commit:  (use "git add <file>..." to update what will be committed)  (use "git restore <file>..." to discard changes in working directory)  modified: test\_git.txt # 有修改  no changes added to commit (use "git add" and/or "git commit -a")  # **修改文件后**，提示modified: test\_git.txt 文件有修改  snowc@SNOW-CITY MINGW64 /f/WorkSpace/geekbang/learn\_git (master)  $ **git log**  commit 4481e5b71287dbd980fc4c32570dcdfebc558079 (HEAD -> master)  Author: snowcitygame <snowcitygame@qq.com>  Date: Sat Jun 27 01:54:11 2020 +0800  add message  commit ccac291be3b87075766cd607f4096dd3194fb426  Author: snowcitygame <snowcitygame@qq.com>  Date: Sat Jun 27 01:35:30 2020 +0800  new empty file test\_git.txt  #显示提交历史  snowc@SNOW-CITY MINGW64 /f/WorkSpace/geekbang/learn\_git (master) |
| **工作区 working directory：**  新建、复制、修改文件时在该区域  提示Untracked files:#未跟踪的文件  使用 **git add** 命令文件进入暂存区  **暂存区 staging area：**  暂存文件  提示为提交和取消暂存（use "git rm --cached <file>..." to unstage）命令  使用 **git commit -m “注释”** 命令进行提交 放入Git仓库  **Git仓库 repository** |