

# git的使用

## 1.版本控制

### 1.1 备份文件

- 类似于网盘备份
- 我们的代码也需要备份。修改完了以后提交给版本库进行保管，哪一天代码没了也可以找回来。

### 1.2 记录历史

- 比如我们打游戏就要 存档 ，万一挂了还可以从上个存档的地方重玩。
- 和网盘不同，网盘保留的是最新的状态，历史的记录 都没有了，修改的记录也都找不回来了
- 网盘无法知道文件里的某行 代码 是何人在哪个时间添加进去的

### 1.3 回到过去

## git的使用

### 1.版本控制

- 1.1 备份文件
- 1.2 记录历史
- 1.3 回到过去
- 1.4 多端共享
- 1.5 团队协作

### 2.什么是git

- 2.1 svn和git的区别

### 3.git的安装

- 3.1 windows下安装
- 3.2 mac下安装

### 4. 配置git用户和邮箱

### 5.初始化git

### 6.git中的三个区

### 7.git diff

- 7.1 工作区和暂存区
- 7.2 暂存区和历史区
- 7.3 工作区和版本库

### 8. 撤销

- 8.1 撤销回git add的内容
- 8.2 撤回文件

- 如果我有一天不小心删除了某个文件，我们可以通过历史备份找回来

## 1.4 多端共享

- Git仓库可以通过PC端、Android、IOS 移动端 等各个终端访问
- 可以 随时随地 修改代码,公司没干完的工作回家接着干

## 1.5 团队协作

- 多个人或团队 合作 编写一个项目
- 合并代码处理 冲突

# 2.什么是git

- 为了告别手动方式管理Linux代码,并且符合开源和免费,Linus花了 两周时间 自己用 C 写了一个分布式版本控制系统，这就是Git
- Git迅速成为最流行的分布式版本控制系统，尤其是 2008 年 GitHub 网站上线了，它为开源项目 免费提供Git存储，无数开源项目开始迁移至GitHub，包括jQuery，PHP，Ruby等等。

## 2.1 svn和git的区别

- 1.速度快
- 2.GIT是分分布式的
- 3.无需联网

## 9.删除

### 9.1 删除暂存区和工作区

### 9.2 仅删除缓存区

## 10. 恢复

## git的使用

### 1.版本控制

- 1.1 备份文件
- 1.2 记录历史
- 1.3 回到过去
- 1.4 多端共享
- 1.5 团队协作

### 2.什么是git

- 2.1 svn和git的区别

### 3.git的安装

- 3.1 windows下安装
- 3.2 mac下安装

### 4. 配置git用户和邮箱

### 5.初始化git

### 6.git中的三个区

### 7.git diff

- 7.1 工作区和暂存区
- 7.2 暂存区和历史区
- 7.3 工作区和版本库

### 8. 撤销

- 8.1 撤销回git add的内容
- 8.2 撤回文件

- 4.GIT把内容按元数据方式存储
- 5.强大的分支管理

## 3.git的安装

### 3.1 windows下安装

下载地址 <http://git-scm.com>

### 9.删除

#### 9.1 删除暂存区和工作区

#### 9.2 仅删除缓存区

### 10. 恢复

## git的使用

### 1.版本控制

- 1.1 备份文件
- 1.2 记录历史
- 1.3 回到过去
- 1.4 多端共享
- 1.5 团队协作

### 2.什么是git

- 2.1 svn和git的区别

### 3.git的安装

- 3.1 windows下安装
- 3.2 mac下安装

### 4. 配置git用户和邮箱

### 5.初始化git

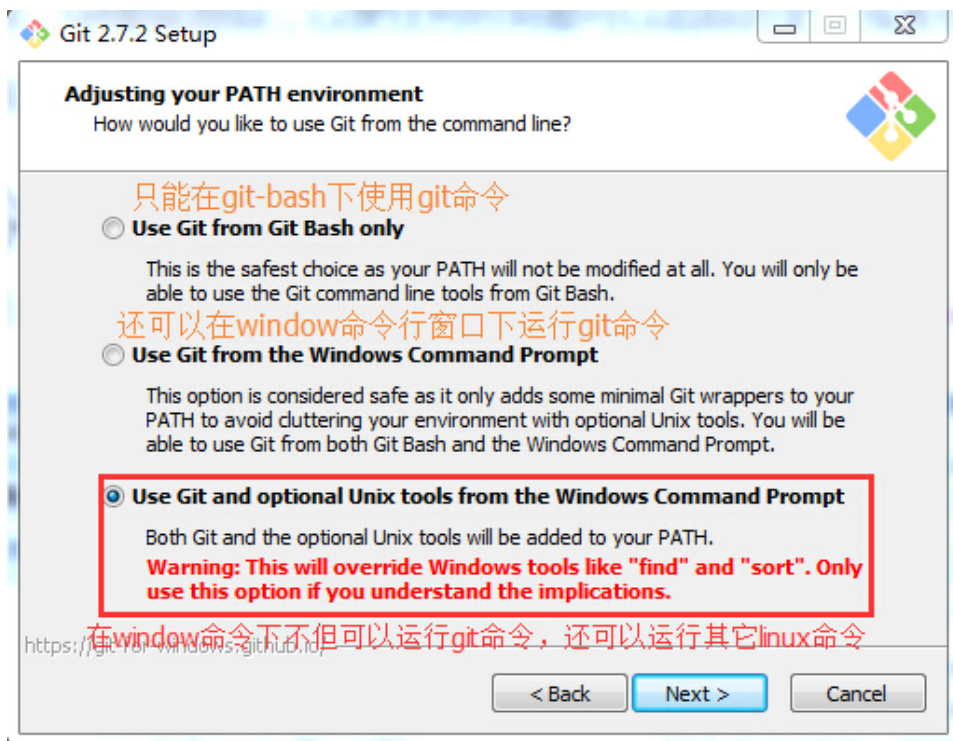
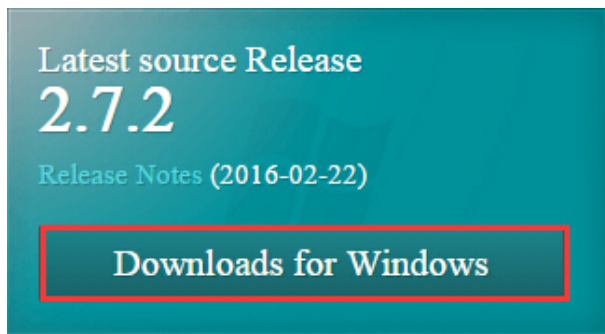
### 6.git中的三个区

### 7.git diff

- 7.1 工作区和暂存区
- 7.2 暂存区和历史区
- 7.3 工作区和版本库

### 8. 撤销

- 8.1 撤销回git add的内容
- 8.2 撤回文件



9.2 仅删除缓存区

## 9.删除

### 9.1 删除暂存区和工作区

### 9.2 仅删除缓存区

## 10. 恢复

## git的使用

### 1.版本控制

- 1.1 备份文件
- 1.2 记录历史
- 1.3 回到过去
- 1.4 多端共享
- 1.5 团队协作

## 2.什么是git

- 2.1 svn和git的区别

## 3.git的安装

- 3.1 windows下安装
- 3.2 mac下安装

## 4. 配置git用户和邮箱

## 5.初始化git

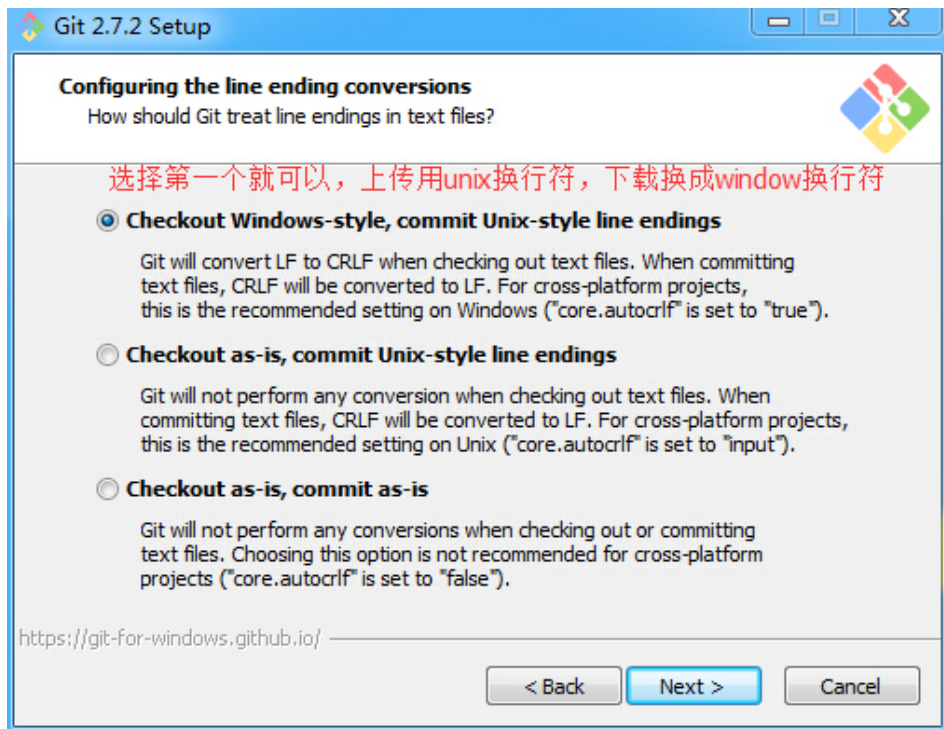
## 6.git中的三个区

## 7.git diff

- 7.1 工作区和暂存区
- 7.2 暂存区和历史区
- 7.3 工作区和版本库

## 8. 撤销

- 8.1 撤销回git add的内容
- 8.2 撤回文件



## 3.2 mac下安装

下载Homebrew<http://brew.sh>



- 拷贝对应脚本到终端下安装HomeBrew
- 然后在终端执行 `brew install git`命令安装\*git

安装xcode会默认下载git

## 9.删除

### 9.1 删除暂存区和工作区

### 9.2 仅删除缓存区

## 10. 恢复

## git的使用

### 1.版本控制

- 1.1 备份文件
- 1.2 记录历史
- 1.3 回到过去
- 1.4 多端共享
- 1.5 团队协作

### 2.什么是git

- 2.1 svn和git的区别

### 3.git的安装

- 3.1 windows下安装
- 3.2 mac下安装

### 4. 配置git用户和邮箱

### 5.初始化git

### 6.git中的三个区

### 7.git diff

- 7.1 工作区和暂存区
- 7.2 暂存区和历史区
- 7.3 工作区和版本库

### 8. 撤销

- 8.1 撤销回git add的内容
- 8.2 撤回文件

## 4. 配置git用户和邮箱

```
$ git config --global user.name "你的github用户名"  
$ git config --global user.email "你的github邮箱"
```

不配置用户名和邮箱的话无法提交,因为git不知道你是谁

查看配置

```
$ git config --global user.name  
$ git config --global user.email
```

查看所有配置

```
$ git config --list
```

## 5. 初始化git

- 先创建一个空目录 , 然后进入此目录
- 点击右键选择Git-Bash打开命令行
- 输入git init命令把这个目录变成Git可以管理的仓库

```
$ git init
```

通过ls -al命令查看所有文件

## 9. 删除

### 9.1 删除暂存区和工作区

### 9.2 仅删除缓存区

## 10. 恢复

## git的使用

### 1. 版本控制

- 1.1 备份文件
- 1.2 记录历史
- 1.3 回到过去
- 1.4 多端共享
- 1.5 团队协作

### 2. 什么是git

- 2.1 svn和git的区别

### 3. git的安装

- 3.1 windows下安装
- 3.2 mac下安装

### 4. 配置git用户和邮箱

### 5. 初始化git

### 6. git中的三个区

### 7. git diff

- 7.1 工作区和暂存区
- 7.2 暂存区和历史区
- 7.3 工作区和版本库

### 8. 撤销

- 8.1 撤销回git add的内容
- 8.2 撤回文件

# 6.git中的三个区

## 工作流

<http://card.mugeda.com/campaigns/56d2c4a0a3664e3308000407/20160304090522/56d97729a3664e9c65000047/index.html>

- 工作区

通过git add 添加到暂存区

```
$ git add '文件名'
```

- 暂存区

特点: 过渡 的作用, 避免 误操作 , 保护 工作区和历史区, 分支处理;

通过git commit 添加到历史区

```
$ git commit -m"注释内容"
```

- 历史区

查看历史状态

```
$ git log
```

修改时通过git status查看当前状态

## 7.git diff

## 9.删除

### 9.1 删除暂存区和工作区

### 9.2 仅删除缓存区

## 10. 恢复

## git的使用

### 1.版本控制

- 1.1 备份文件
- 1.2 记录历史
- 1.3 回到过去
- 1.4 多端共享
- 1.5 团队协作

### 2.什么是git

- 2.1 svn和git的区别

### 3.git的安装

- 3.1 windows下安装
- 3.2 mac下安装

### 4. 配置git用户和邮箱

### 5.初始化git

### 6.git中的三个区

### 7.git diff

- 7.1 工作区和暂存区
- 7.2 暂存区和历史区
- 7.3 工作区和版本库

### 8. 撤销

- 8.1 撤销回git add的内容
- 8.2 撤回文件



不同区的代码比较

## 7.1 工作区和暂存区

```
$ git diff
```

## 7.2 暂存区和历史区

```
$ git diff --cached (--staged)
```

## 7.3 工作区和版本库

```
$ git diff master
```

# 8. 撤销

---

## 8.1 撤销回git add的内容

```
git reset Head "文件名"
```

## 8.2 撤回文件

- 先从缓存区撤销,缓存区无内容,从历史区域撤销

```
$ git checkout "文件名"
```

## 9.删除

### 9.1 删除暂存区和工作区

### 9.2 仅删除缓存区

## 10. 恢复

## git的使用

### 1.版本控制

- 1.1 备份文件
- 1.2 记录历史
- 1.3 回到过去
- 1.4 多端共享
- 1.5 团队协作

### 2.什么是git

- 2.1 svn和git的区别

### 3.git的安装

- 3.1 windows下安装
- 3.2 mac下安装

### 4. 配置git用户和邮箱

### 5.初始化git

### 6.git中的三个区

### 7.git diff

- 7.1 工作区和暂存区
- 7.2 暂存区和历史区
- 7.3 工作区和版本库

### 8. 撤销

- 8.1 撤销回git add的内容
- 8.2 撤回文件



有的时候我们希望提交时合并到上一次的提交 `git commit --amend`

## 9. 删除

### 9.1 删除暂存区和工作区

删除暂存区中的内容,并且保证工作区中的内容已经不存在

```
$ git rm "文件名"
```

若本地文件存在则不能删除,需要通过-f参数删除

### 9.2 仅删除缓存区

```
$ git rm --cached "文件名"
```

## 10. 恢复

### 10.1 恢复某个版本文件

```
$ git checkout commit_id filename 某个文件
```

### 9. 删除

#### 9.1 删除暂存区和工作区

#### 9.2 仅删除缓存区

### 10. 恢复

## git的使用

### 1. 版本控制

- 1.1 备份文件
- 1.2 记录历史
- 1.3 回到过去
- 1.4 多端共享
- 1.5 团队协作

### 2. 什么是git

- 2.1 svn和git的区别

### 3. git的安装

- 3.1 windows下安装
- 3.2 mac下安装

### 4. 配置git用户和邮箱

### 5. 初始化git

### 6. git中的三个区

### 7. git diff

- 7.1 工作区和暂存区
- 7.2 暂存区和历史区
- 7.3 工作区和版本库

### 8. 撤销

- 8.1 撤销回git add的内容
- 8.2 撤回文件

## 10.2 通过版本id恢复

```
$ git reset --hard commit_id
```

## 10.3 恢复未来

查看当时回滚时的版本

```
$ git reflog
```

## 10.4 快速版本回退

```
$ git reset --hard HEAD^  
$ git reset --hard HEAD~3
```

# 11. 分支管理

---

## 11.1 创建分支

```
$ git branch dev
```

## 11.2 切换分支

```
$ git checkout dev
```

## 9.删除

### 9.1 删除暂存区和工作区

### 9.2 仅删除缓存区

## 10. 恢复

## git的使用

### 1.版本控制

- 1.1 备份文件
- 1.2 记录历史
- 1.3 回到过去
- 1.4 多端共享
- 1.5 团队协作

### 2.什么是git

- 2.1 svn和git的区别

### 3.git的安装

- 3.1 windows下安装
- 3.2 mac下安装

### 4. 配置git用户和邮箱

### 5.初始化git

### 6.git中的三个区

### 7.git diff

- 7.1 工作区和暂存区
- 7.2 暂存区和历史区
- 7.3 工作区和版本库

### 8. 撤销

- 8.1 撤销回git add的内容
- 8.2 撤回文件

## 11.3 创建并切换分支

```
$ git checkout -b dev
```

## 11.4 查看分支

```
$ git branch
```

## 11.5 合并分支

```
$ git merge dev
```

## 11.6 删除分支

```
$ git branch -d dev
```

## 11.7 查看提交信息

```
git log --oneline --graph --decorate --all
```

## 11.8 存储工作区

```
$ git stash list
```

## 9.删除

### 9.1 删除暂存区和工作区

### 9.2 仅删除缓存区

## 10. 恢复

## git的使用

### 1.版本控制

- 1.1 备份文件
- 1.2 记录历史
- 1.3 回到过去
- 1.4 多端共享
- 1.5 团队协作

### 2.什么是git

- 2.1 svn和git的区别

### 3.git的安装

- 3.1 windows下安装
- 3.2 mac下安装

### 4. 配置git用户和邮箱

### 5.初始化git

### 6.git中的三个区

### 7.git diff

- 7.1 工作区和暂存区
- 7.2 暂存区和历史区
- 7.3 工作区和版本库

### 8. 撤销

- 8.1 撤销回git add的内容
- 8.2 撤回文件

## 11.9 删除存储的历史

```
$ git stash apply  
$ git stash drop
```

```
$ git stash pop
```

## 11.10 rebase

```
$ git rebase dev
```

## 11.11 cherry-pick

```
$ git rebase dev
```

## 11.12 创建标签

```
$ git tag v1.0
```

## 11.13 查看标签

```
git show v1.0
```

# 12. 同步远程仓库

## 9.删除

### 9.1 删除暂存区和工作区

### 9.2 仅删除缓存区

## 10. 恢复

## git的使用

### 1.版本控制

- 1.1 备份文件
- 1.2 记录历史
- 1.3 回到过去
- 1.4 多端共享
- 1.5 团队协作

### 2.什么是git

- 2.1 svn和git的区别

### 3.git的安装

- 3.1 windows下安装
- 3.2 mac下安装

### 4. 配置git用户和邮箱

### 5.初始化git

### 6.git中的三个区

### 7.git diff

- 7.1 工作区和暂存区
- 7.2 暂存区和历史区
- 7.3 工作区和版本库

### 8. 撤销

- 8.1 撤销回git add的内容
- 8.2 撤回文件

## 12.1 gitHub

- 注册账号
- 新建项目

## 12.2 添加远程仓库

```
$ git remote add origin "地址"
```

## 12.3 添加忽略文件

```
$ touch .gitignore  
$ echo .DS_Store  
$ echo node_modules  
$ echo .idea
```

## 12.4 推送代码

```
$ git push origin master
```

## 12.5 查看

```
$ git remote 查看名字  
$ git remote -v 查看地址
```

## 9.删除

### 9.1 删除暂存区和工作区

### 9.2 仅删除缓存区

## 10. 恢复

## git的使用

### 1.版本控制

- 1.1 备份文件
- 1.2 记录历史
- 1.3 回到过去
- 1.4 多端共享
- 1.5 团队协作

### 2.什么是git

- 2.1 svn和git的区别

### 3.git的安装

- 3.1 windows下安装
- 3.2 mac下安装

### 4. 配置git用户和邮箱

### 5.初始化git

### 6.git中的三个区

### 7.git diff

- 7.1 工作区和暂存区
- 7.2 暂存区和历史区
- 7.3 工作区和版本库

### 8. 撤销

- 8.1 撤销回git add的内容
- 8.2 撤回文件

# 13.代码的合并

## 13.1 git fetch

```
$ git fetch
```

拉取过来手动合并

```
$ git diff master origin/master  
$ git merge origin/master
```

## 13.2 git pull

拉取并合并

```
git pull
```

# 14. 作业提交流程

## 14.1 第一次交作业流程(组长)

- fork 珠峰培训讲师的作业仓库
- 把自己的仓库 下载 到本地
- 把自己的作业 上传 到自己的github仓库
- 发 pull request 给讲师

## 9.删除

### 9.1 删除暂存区和工作区

### 9.2 仅删除缓存区

## 10. 恢复

## git的使用

### 1.版本控制

- 1.1 备份文件
- 1.2 记录历史
- 1.3 回到过去
- 1.4 多端共享
- 1.5 团队协作

### 2.什么是git

- 2.1 svn和git的区别

### 3.git的安装

- 3.1 windows下安装
- 3.2 mac下安装

### 4. 配置git用户和邮箱

### 5.初始化git

### 6.git中的三个区

### 7.git diff

- 7.1 工作区和暂存区
- 7.2 暂存区和历史区
- 7.3 工作区和版本库

### 8. 撤销

- 8.1 撤销回git add的内容
- 8.2 撤回文件

- 添加组员账号

## 14.2 第二次交作业流程(组员)

- 克隆组长项目
- 先 拉取 组长仓库最新代码
- 将自己的作业 ( 放入对应的文件夹 )
- git add/git commit
- 再push前拉取组长仓库最新代码
- push到组长仓库

## 14.3 第二次交作业流程(组长)

- 在本地 增加 讲师仓库
- 拉取 自己的最新代码
- 拉取 老师的最新的代码,如果冲突需要解决冲突
- 把自己的作业 拷贝进去(如果组长没有要提交的东西可省略)
- 把自己的作业上 传到 自己的github仓库(如果组长没有要提交的东西可省略)
- 发 pull request 给讲师

参考教程<http://school.zhufengpeixun.cn/course/31>

### 9.删除

#### 9.1 删除暂存区和工作区

#### 9.2 仅删除缓存区

### 10. 恢复

## git的使用

### 1.版本控制

- 1.1 备份文件
- 1.2 记录历史
- 1.3 回到过去
- 1.4 多端共享
- 1.5 团队协作

### 2.什么是git

- 2.1 svn和git的区别

### 3.git的安装

- 3.1 windows下安装
- 3.2 mac下安装

### 4. 配置git用户和邮箱

### 5.初始化git

### 6.git中的三个区

### 7.git diff

- 7.1 工作区和暂存区
- 7.2 暂存区和历史区
- 7.3 工作区和版本库

### 8. 撤销

- 8.1 撤销回git add的内容
- 8.2 撤回文件