wuming_zhuanzhu L25 2017 -04 -26 10:54 Git
47 0

“ Git 实战教程” 实验报告

高级技能

Git用法

标签（空格分隔）： git

> 这个笔记不算高级技能章节的实验报告，这个是我学习git这门课程的笔记总结，笔记内容没有面面俱到，只把常用的内容记录下来，难懂的内容我也直接跳过了。

[TOC]

git配置

```
git config --global <配置名称> <配置的值>
```

`--global` 表示全局配置，不带 `--global` 选项为当前目录下创建.git/confg

正常工作流程及操作

初始化仓库 `git init`
创建或修改文件
`git add` 添加新创建或修改的文件到本地缓存区
`git commit` 命令提交到本地代码块
`git push` 可选，把代码推送到远程仓库

`git diff --cached` 查看缓存区中哪些文件被修改

`git commit` `-a` 参数将所有没有加到缓存区的修改也一起提交，但不会添加新建的文件

`git status` 查看状态

`git rm` 删除缓存文件，提交后删除仓库中对应的文件

`git push origin master`

分支与合并

分支

`git branch` 不加分支名，查看分支； `git branch branch1` 加分支名，创建分支

`git checkout 分支名`

合并

`git merge` `-m` 合并注释

`git reset --hard HEAD^` 撤销一个合并

git日志

`git log`

`git help log`

小结

git confg：配置相关信息

git clone：复制仓库

git init：初始化仓库

git add：添加更新内容到索引中

git dif：比较内容

git status：获取当前项目状况

git commit：提交

git branch：分支相关

git checkout：切换分支

git merge：合并分支

git reset：恢复版本

git log：查看日志

中级技能上

忽略某些文件

`.gitingore`

rebase变基

中级技能下

追踪分支

git grep 进行搜索

撤销操作

- `git reset` 修复未提交文件中的错误（重置） `git reset --hard HEAD`
- `git checkout` 恢复一个文件 `git checkout -- hello.py`
- `git revert` 修复已提交文件中的错误 `git revert HEAD` 撤销最近的一个提交
- `git commit --amend` 修改提交注释

维护git，保证良好的性能和可靠性

- 压缩 `git gc`
- 检查一致性 `git fsck`

公共仓库

建立一个公共仓库

“裸git仓库" -- 即只有'.git'目录里的内容,没有任何签出(checked out)的文件

`git clone --bare project project.git`

把 `project.git` 目录拷贝到托管的主机上

通过git协议导出git仓库
通过http协议导出git仓库

私有仓库

只要把“git裸仓库"放到任何一个可以通过ssh访问的目录
通过ssh协议访问仓库

高级技能

1. 创建新的空分支

```
git symbolic-ref HEAD refs/heads/newbranch
rm .git/index
git clean -fdx
<do work>
git add your files
git commit -m 'Initial commit'
```

2. 修改历史

`git filter-branch` 是修改大量提交的好方法

高级分支与合并

`git diff` 合并confict协助
多路合并 `git merge scott/master rick/master tom/master`
子树 `不懂`
查找问题的利器 `git bisect`
`git blame` 查看文件的每个部分是谁修改的
向项目提交补丁 `git format-patch origin`
导入补丁 `git am -3 patchname.patch` `-3选项会让git执行合并操作`
    解决冲突后，不需要创建一个新的提交，只需运行 `git am --resolved`

定制git

更改编辑器 `git config --global cor.editor vim`

添加别名 `git config --global alias.last 'cat-file commit HEAD'`

```
$ git last
$ git cat-file commit HEAD
```

添加颜色

```
$ git config color.branch auto
$ git config color.diff auto
$ git config color.interactive auto
$ git config color.status auto
把颜色全部打开
$ git config color.ui true
```

提交模板

```
$ git config commit.template '/etc/git-commit-template'
```

日志格式

```
$ git config format.pretty oneline
```

0