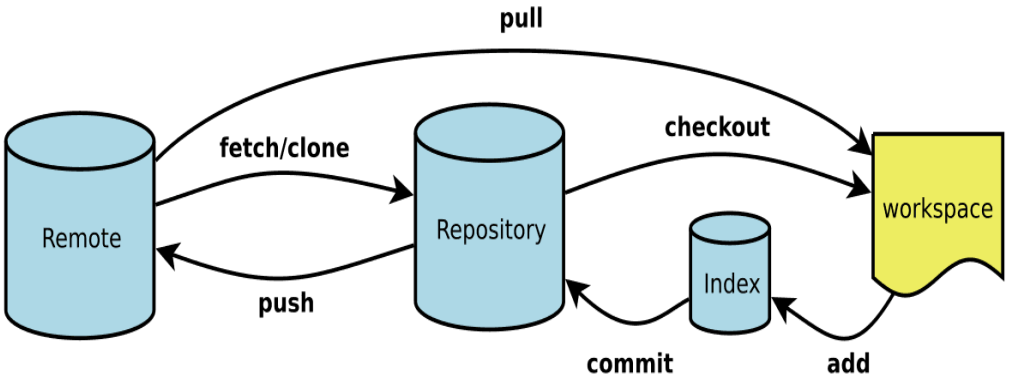
git常用命令指南

首先，要知道的是在git中有以下这几个区域
工作区（workspace）：可以理解为这就是你当前看到的目录、文件
暂存区（index）：git add 以后文件将会存在这里，是介于workspace和Repository之间的一个存在
本地仓库（Repository）：每个人都有自己的一个本地仓库，在本地也可以进行版本控制，同时，本地仓库连接着远程仓库
远程仓库（Remote）：保存着我们代码的服务器，大家各自把本地仓库的代码上传至远程仓库，进行集中的版本控制



下面总结了比较常用的一些git命令，可以结合上面张图进行理解

创建一个本地仓库

有下列三种方法可以创建一个代码仓库

- 在当前目录下创建一个仓库

```
git init
```

- 新建一个目录，将其初始化为git代码库

```
git init [project-name]
```

- 若远程已经有项目了，我们可以在想要存放该项目的目录下执行

```
git clone [url]
```

下载一个远程项目到本地

切换/创建分支

从远程下载一个项目到本地后，本地仓库通常是在master分支上的，我们通常是在其他分支上进行开发
第一步：我们通过以下命令查看分支状态

```
git branch -a
```

查看当前分支以及远程分支
第二步：切换分支

```
git checkout [branch]
```

这条命令的含义如下：

- 如果本地有该分支，那么就会顺利地切换到这个分支上
- 如果本地没有该分支，执行该命令将会检出远程分分支，然后在本地创建一个同名分支，并创建与远程上游的跟踪
- 如果本地和远程都没有这个分支，那么该命令执行失败

第三步：查看跟踪
建立跟踪后，通过这条命令就可以看到本地分支和远程分支对应的情况了

```
git branch -vv
```

建立跟踪的作用在于当你使用git pull/push/fetch时如果不指定远端分支，就会使用该跟踪对应的远程分支
或者也可以自己创建分支，再建立跟踪
第一步：
在本地创建分支

```
git branch [branch]
```

该命令是在当前分支下执行的
第二步：
建立跟踪

```
git branch --set-upstream [localbranch] origin/[remotebranch]
```

从远程拉取分支

```
git pull [remote] [branch]
```

从远程获取最新的代码并和本地分支合并
上述命令相当于执行了以下两条命令

1. git fetch 从远程获取最新代码到本地，但是不会合并
2. git merge合并代码

所以，在实际使用中，更加安全的方式是

1. git fetch
2. git diff 查看暂存区和工作区的区别
3. git merge

我们可以用git diff看看有什么变化然后再选择是否合并

提交本地代码到远程仓库

通常我们使用频率最高的下面的一套指令，用于提交修改并上传到远程仓库
在工作区修改完文件后，我们要提交修改到远程，用下列步骤

第一步：git add 提交工作区中的修改到暂存区

```
git add . 或者 git add <file>
```

git add . 提交所有工作区中的修改到暂存区
git add <file> 提交该文件在工作区中的修改到暂存区

第二步：
git commit 提交暂存区中的内容到本地仓库

```
git commit -m "xxx" 或者 git commit -m "xxx" <file>
```

git commit -m "本次修改的内容" 将暂存区中的所有内容提交到本地仓库
git commit -m "本次修改的内容" <file> 将暂存区中的某文件提交到本地仓库

第三步：
为了避免当前的代码不是最新的，在push之前执行该命令

```
git pull --rebase
```

合并代码前下载最新代码
该命令执行了以下内容：

a. 把本地 仓库 从上次 pull 之后的变更暂存起來
b. 恢复到上次 pull 时的状态
c. 合并远端的变更到本地
d. 最后再合并刚刚暂存下來的本地变更

第四步：git push 推送本地分支到远程仓库

```
git push [remote]
```

推送当前分支到远程仓库
我们通常push到该分支
git push origin HEAD:refs/for/XXXXX
eg.

```
git push origin HEAD:refs/for/release_7.05.2
```

补充：在上面三步每一步执行前后都可以用git status看一下当前的状态，不仅可以看到你的改动，还会提示相关操作的git命令，非常实用

如果一切顺利的话，我们通过这四步就能成功提交我们的代码到远程仓库了
但经常会发生冲突，导致我们提交失败

合并分支

假设我们有分支a和分支b要进行合并
第一步：切换到a分支

```
 git checkout a
```

第二步：合并前下载最新代码

```
git pull --rebase
```

第三步：切换到b分支

```
git checkout b
```

第四步： 合并前下载最新代码

```
git pull --rebase
```

第五步：将a分支合并到b分支（在b分支下执行）

```
git merge a
```

如果没有冲突发生，就成功合并了

解决冲突

不管是合并或是push，冲突时常发生，关于冲突，最常见的就是内容冲突
发生冲突后，通过git status我们可以看到有哪些文件存在冲突，然后手动解决
git会用 <<<<<<<，=======，>>>>>>>标记出不同分支的内容，可以修改冲突文件中这些标记存在的地方来解决冲突
但是如果冲突有很多的话，使用SourceTree这样的图形界面来解决更加方便
具体可以查看http://conf.ctripcorp.com/pages/viewpage.action?pageId=140132366
当我们把冲突解决后
执行git commit -m "fix conflict"来提交修复的冲突

撤销

撤销工作区修改

如果我们想撤销在工作区的修改，让工作区回到文件修改前的状态
第一步：查看当前状态

```
git status
```

进行git status查看后发现
git会有如下提示

```
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard chang
```

```
es in working directory)
          modified:
```

第二步：撤销工作区修改

```
git checkout -- <file>
```

第三步 ：查看当前状态

```
git status
```

如果执行成功，将会看到上面的提示内容下的modified将不会出现该文件

撤销暂存区修改

也就是在文件修改后进行了git add，文件的修改这时候已经提交到了暂存区，将该文件从暂存区撤回
第一步:查看当前状态

```
git status
```

进行git status查看后发现
git会有如下提示

```
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)
          modified:
```

第二步：撤销暂存区修改

```
git reset HEAD <file>
```

第三步：查看当前状态

```
git status
```

如果执行成功，将会看到上面的提示内容下的modified将不会出现该文件

撤销本地仓库的修改

也就是撤销git commit操作，进行git commit之后想要撤销某一次的
第一步：查看提交日志

```
git log
```

输入git log后会显示出现从最近到最远的提交信息
第二步：回滚
在Git中，用HEAD表示当前版本，也就是最新的提交的（用git log查看到的第一条），上一个版本就是HEAD^或者HEAD~1，上上一个版本就是HEAD^^或者HEAD~2，往上100个版本可写成HEAD~100
例如回退到上一个版本

```
git reset --hard HEAD^
```

或者

```
git reset --hard commit-id      //commit-id为上一版本的commit-id
```

--hard参数的含义：重设暂存区和工作区，自从<commit>以来在工作区中的任何改变都被丢弃，并把HEAD指向<commit>。

回滚以后再回去

在上面的撤销操作成功后，回退到了某个版本，但是又后悔了，想恢复到新版本
这时候使用git log是找不到新版本的commit id的，我们可以用另一条命令进行查询
第一步：查询

```
git reflog
```

会出现你每一次操作的信息，找到之前commit的那条的id

```
52c861 HEAD@{0}: reset: moving to HEAD~1
1f5da2e HEAD@{1}: commit: 修改国内保险页部分标签
852c861 HEAD@{2}: checkout: moving from dev to cde
```

例如这里我需要回退的版本是第二条，commit-id就是1f5da2e
第二步：恢复新版本

```
git reset --hard commit-id
```

撤销push操作

当你由于不小心将代码push到远程了，但是发现有问题需要回退
第一步：找到你想回去的版本的commit-id

```
git log
```

通过该命令找到你要的commit-id
第二步：回滚

```
git revert commit-id
```

下面这篇文章提到了大量的撤销操作，有需要可以查看
http://www.cnblogs.com/allencelee/p/5603914.html

常用查看命令

- 查看本地分支

```
git branch
```

- 查看远程分支

```
git branch -r
```

- 查看所有分支（本地和远程）

```
git branch -a
```

- 查看变更文件

```
git status
```

- 查看当前分支的历史版本

```
git log
```

- 查看暂存区与工作区区别

```
git diff
```