丢弃工作区的修改

git checkout -- file

我先再添加一个checkoutTest.docx文件，并给他写上一行：git checkout

然后使用git add、git commit指令把它添加到版本库

我再给这个文件添加一行:delete it

使用git checkout -- checkoutTest.docx

再打开这个文件，又只剩下一行git checkout了。

之后我再给这个文件添加一行 in stage,并使用git add把它提交到暂存区

现在我再给它添加一行：just in the work并使用git checkout,

点开文件，只剩下git checkout 和in stage

命令git checkout -- readme.txt意思就是，把readme.txt文件在工作区的修改全部撤销，这里有两种情况：

一种是readme.txt自修改后还没有被放到暂存区，现在，撤销修改就回到和版本库一模一样的状态；

一种是readme.txt已经添加到暂存区后，又作了修改，现在，撤销修改就回到添加到暂存区后的状态。

总之，就是让这个文件回到最近一次git commit或git add时的状态。

git checkout -- file命令中的--很重要，没有--，就变成了“切换到另一个分支”的命令，我们在后面的分支管理中会再次遇到git checkout命令。

现在使用git status查看情况

$ git status

On branch master

Changes to be committed:

(use "git reset HEAD <file>..." to unstage)

modified: checkoutTest.docx

checkoutTest.docx还在暂存区，并提示可以用git reset HEAD file命令撤销暂存区的文件

丢弃暂存区的修改

用命令git reset HEAD file可以把暂存区的修改撤销掉（unstage），重新放回工作区：

打开文件，两行文字还在，使用git status

$ git status

On branch master

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git checkout -- <file>..." to discard changes in working directory)

modified: checkoutTest.docx

已经不在暂存区啦

丢弃版本库的修改

git reset -- hard HEAD^或者

git reset -- hard commit\_id

* 穿梭前，用git log可以查看提交历史，以便确定要回退到哪个版本。
* 要重返未来，用git reflog查看命令历史，以便确定要回到未来的哪个版本。

删除版本库的文件

git rm 并commit到版本库