我们已经成功地添加并提交了一个readme.txt文件，现在，是时候继续工作了，于是，我们继续修改readme.txt文件.

现在，运行git status命令看看结果

admin@zebin MINGW64 /d/BlueTestTry/bluetooth (master)

$ git status

On branch master

Your branch is ahead of 'origin/master' by 1 commit.

(use "git push" to publish your local commits)

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: readme.txt

no changes added to commit (use "git add" and/or "git commit -a")

admin@zebin MINGW64 /d/BlueTestTry/bluetooth (master)

$ git diff

**diff --git a/readme.txt b/readme.txt**

**index 070bf74..c53dade 100644**

**--- a/readme.txt**

**+++ b/readme.txt**

@@ -1,2 +1,2 @@

-Git is a version control system

+Git is a distributed version control system^M

Git is free software

\ No newline at end of file

admin@zebin MINGW64 /d/BlueTestTry/bluetooth (master)

$ git add readme.txt

admin@zebin MINGW64 /d/BlueTestTry/bluetooth (master)

$ git status

On branch master

Your branch is ahead of 'origin/master' by 1 commit.

(use "git push" to publish your local commits)

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

modified: readme.txt

git status告诉我们，将要被提交的修改包括readme.txt，下一步，就可以放心地提交了

admin@zebin MINGW64 /d/BlueTestTry/bluetooth (master)

$ git commit -m "add distributed"

[master 20c0b88] add distributed

1 file changed, 1 insertion(+), 1 deletion(-)

git log命令显示从最近到最远的提交日志，我们可以看到3次提交，最近的一次是append GPL，上一次是add distributed，最早的一次是wrote a readme file。如果嫌输出信息太多，看得眼花缭乱的，可以试试加上--pretty=oneline参数：

admin@zebin MINGW64 /d/BlueTestTry/bluetooth (master)

$ git log --pretty=oneline

dd1fcdda882fe8b95e519f9af2b39b9a8ea883be (HEAD -> master) append GPL

20c0b882831bf684dce1ef8579a82e063437e0d9 add distributed

2557423f9e1e3903eaf76283d761049cad8af4bf wrote an new readme file

好了，现在我们启动时光穿梭机，准备把readme.txt回退到上一个版本，也就是add distributed的那个版本，怎么做呢？

首先，Git必须知道当前版本是哪个版本，在Git中，用HEAD表示当前版本，也就是最新的提交1094adb...（注意我的提交ID和你的肯定不一样），上一个版本就是HEAD^，上上一个版本就是HEAD^^，当然往上100个版本写100个^比较容易数不过来，所以写成HEAD~100。

现在，我们要把当前版本append GPL回退到上一个版本add distributed，就可以使用git reset命令

admin@zebin MINGW64 /d/BlueTestTry/bluetooth (master)

$ git reset --hard HEAD^

HEAD is now at 20c0b88 add distributed

admin@zebin MINGW64 /d/BlueTestTry/bluetooth (master)

$ git log --pretty=oneline

20c0b882831bf684dce1ef8579a82e063437e0d9 (HEAD -> master) add distributed

2557423f9e1e3903eaf76283d761049cad8af4bf wrote an new readme file

--hard参数有啥意义？--hard会回退到上个版本的已提交状态，而--soft会回退到上个版本的未提交状态，--mixed会回退到上个版本已添加但未提交的状态。现在，先放心使用--hard。

admin@zebin MINGW64 /d/BlueTestTry/bluetooth (master)

$ cat readme.txt

Git is a distributed version control system

Git is free software

最新的那个版本append GPL已经看不到了！好比你从21世纪坐时光穿梭机来到了19世纪，想再回去已经回不去了，肿么办？

办法其实还是有的，只要上面的命令行窗口还没有被关掉，你就可以顺着往上找啊找啊，找到那个append GPL的commit id是1094adb...，于是就可以指定回到未来的某个版本：

admin@zebin MINGW64 /d/BlueTestTry/bluetooth (master)

$ git reset --hard dd1fc

HEAD is now at dd1fcdd append GPL

在Git中，总是有后悔药可以吃的。当你用$ git reset --hard HEAD^回退到add distributed版本时，再想恢复到append GPL，就必须找到append GPL的commit id。Git提供了一个命令git reflog用来记录你的每一次命令

admin@zebin MINGW64 /d/BlueTestTry/bluetooth (master)

$ git reflog

20c0b88 (HEAD -> master) HEAD@{0}: reset: moving to 20c0b

dd1fcdd HEAD@{1}: reset: moving to dd1fc

20c0b88 (HEAD -> master) HEAD@{2}: reset: moving to HEAD^

dd1fcdd HEAD@{3}: commit: append GPL

20c0b88 (HEAD -> master) HEAD@{4}: commit: add distributed

2557423 HEAD@{5}: commit: wrote an new readme file

d0abb08 (origin/master, origin/HEAD) HEAD@{6}: clone: from git@github.com:ZebinGao/bluetooth.git

为什么Git比其他版本控制系统设计得优秀，因为Git跟踪并管理的是修改，而非文件。

你看，我们前面讲了，Git管理的是修改，当你用git add命令后，在工作区的第一次修改被放入暂存区，准备提交，但是，在工作区的第二次修改并没有放入暂存区，所以，git commit只负责把暂存区的修改提交了，也就是第一次的修改被提交了，第二次的修改不会被提交。

提交后，用git diff HEAD -- readme.txt命令可以查看工作区和版本库里面最新版本的区别：

admin@zebin MINGW64 /d/BlueTestTry/bluetooth (master)

$ git diff HEAD -- readme.txt

**diff --git a/readme.txt b/readme.txt**

**index d7a4c3c..be13f15 100644**

**--- a/readme.txt**

**+++ b/readme.txt**

@@ -1,4 +1,4 @@

Git is a distributed version control system

Git is free software distributed under the GPL

Git has a mutable index called stage.

-Git tracks changes.

\ No newline at end of file

+Git tracks changes of files.

\ No newline at end of file