#### 1、临时保存修改

当你接到一个修复一个代号101的bug的任务时，很自然地，你想创建一个分支issue-101来修复它，但是，等等，当前正在dev上进行的工作还没有提交：



并不是你不想提交，而是工作只进行到一半，还没法提交，预计完成还需1天时间。但是，必须在两个小时内修复该bug，怎么办？

Git提供了一个stash功能，可以把当前工作现场“储藏”起来，等以后恢复现场后继续工作：

git stash

现在，用git status查看工作区，就是干净的（除非没有被Git管理的文件），因此可以放心地创建分支来修复bug。

#### 2、切换到需要修复的分支并新建修复bug的分支

首先确定要在哪个分支上修复bug，假定需要在master分支上修复，就从master创建临时分支：

$ git checkout master //切换到master分支

$ git checkout -b hotfix\_20200615 //新建修复bug的分支

#### 3、修复bug并提交代码

在hotfix\_20200615分支上修改代码，并提交。

$ git add .gitignore

$ git commit -m "hotfix\_20200615"

#### 4、修复完后，切换到master分支，将修复bug的分支合并到master

$ git checkout master

$ git merge --no-ff -m "merged hotfix\_20200615" hotfix\_20200615

#### 5、切回到dev继续开发

$ git switch dev

$ git status

工作区是干净的，刚才的工作现场存到哪去了？用git stash list命令看看：

$ git stash list

stash@{0}: WIP on dev: f52c633 add merge

工作现场还在，Git把stash内容存在某个地方了，但是需要恢复一下，有两个办法：

一是用git stash apply恢复，但是恢复后，stash内容并不删除，你需要用git stash drop来删除；

另一种方式是用git stash pop，恢复的同时把stash内容也删了：

$ git stash pop

On branch dev

Changes to be committed:

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

new file: hello.py

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: readme.txt

Dropped refs/stash@{0} (5d677e2ee266f39ea296182fb2354265b91b3b2a)