## 1.git diff与git apply：生成的标准patch

我们可以首先用git diff制作一个patch。本文示例的工作目录里最初有一个文件a，内容是“This is the file a.”，放置在master分支中。为了修改代码，我们一般的做法是建立一个新分支：

sweetdum@sweetdum-ASUS:~/GitEx$ git branch Fix

sweetdum@sweetdum-ASUS:~/GitEx$ git checkout Fix

Switched to branch 'Fix'

接下来我们在a文件里面追加一行，然后执行git diff。

sweetdum@sweetdum-ASUS:~/GitEx$ echo 'Fix!!!'>>a

sweetdum@sweetdum-ASUS:~/GitEx$ git diff

diff --git a/a b/a

index 4add65f..0d295ac 100644

--- a/a

+++ b/a

@@ -1 +1,2 @@

This is the file a.

+Fix!!!

我们看到了Git diff的输出，这是一个非常典型的Patch式diff。这样我们可以直接把这个输出变为一个Patch：

sweetdum@sweetdum-ASUS:~/GitEx$ git commit -a -m "Fix"

[Fix b88c46b] Fix

1 files changed, 1 insertions(+), 0 deletions(-)

sweetdum@sweetdum-ASUS:~/GitEx$ git diff master > patch

sweetdum@sweetdum-ASUS:~/GitEx$ git checkout master

Switched to branch 'master'

我们现在有一个patch文件，并且签出了master，接下来我们可以使用git apply来应用这个patch。当然了，实际应用中，我们不会这样在一个分支建patch，到另一个分支去应用，因为只有merge一下就好了。我们现 在权当没有这个Fix分支。一般情况下，为了保护master，我们会建立一个专门处理新交来的patch的分支：

sweetdum@sweetdum-ASUS:~/GitEx$ git branch PATCH

sweetdum@sweetdum-ASUS:~/GitEx$ git checkout PATCH

Switched to branch 'PATCH'

sweetdum@sweetdum-ASUS:~/GitEx$ git apply patch

sweetdum@sweetdum-ASUS:~/GitEx$ git commit -a -m "Patch Apply"

[PATCH 9740af8] Patch Apply

1 files changed, 1 insertions(+), 0 deletions(-)

看，现在我们在PATCH分支中应用了这个补丁，我们可以把PATCH分支和Fix比对一下，结果肯定是什么也没有，说明PATCH分支和Fix分支完全一样。patch应用成功。即使有多个文件git diff 也能生成一个patch。

## 2.git format-patch与git am：生成的git专用补丁。

我们同样用上面那个例子的工作目录，这次，我们在Fix分支中的a添加了新行之后，用git format-patch生成一个patch。

sweetdum@sweetdum-ASUS:~/GitEx$ git checkout Fix

Switched to branch 'Fix'

sweetdum@sweetdum-ASUS:~/GitEx$ echo 'Fix!!!'>>a

sweetdum@sweetdum-ASUS:~/GitEx$ git commit -a -m "Fix1"

[Fix 6991743] Fix1

1 files changed, 1 insertions(+), 0 deletions(-)

sweetdum@sweetdum-ASUS:~/GitEx$ git format-patch -M master

0001-Fix1.patch

git format-patch的-M选项表示这个patch要和那个分支比对。现在它生成了一个patch文件，我们看看那是什么：

sweetdum@sweetdum-ASUS:~/GitEx$ cat 0001-Fix1.patch

From 6991743354857c9a6909a253e859e886165b0d90 Mon Sep 17 00:00:00 2001

From: Sweetdumplings <linmx0130@163.com>

Date: Mon, 29 Aug 2011 14:06:12 +0800

Subject: [PATCH] Fix1

---

a | 1 +

1 files changed, 1 insertions(+), 0 deletions(-)

diff --git a/a b/a

index 4add65f..0d295ac 100644

--- a/a

+++ b/a

@@ -1 +1,2 @@

This is the file a.

+Fix!!!

--

1.7.4.1

看，这次多了好多东西，不仅有diff的信息，还有提交者，时间等等，仔细一看你会发现，这是个E-mail的文件，你可以直接发送它！这种patch，我们要用git am来应用。

sweetdum@sweetdum-ASUS:~/GitEx$ git checkout master

Switched to branch 'master'

sweetdum@sweetdum-ASUS:~/GitEx$ git branch PATCH

sweetdum@sweetdum-ASUS:~/GitEx$ git checkout PATCH

sweetdum@sweetdum-ASUS:~/GitEx$ git am 0001-Fix1.patch

Applying: Fix1

sweetdum@sweetdum-ASUS:~/GitEx$ git commit -a -m "PATCH apply"

在提交了补丁之后，我们可以再看看目前文件a的情况：

sweetdum@sweetdum-ASUS:~/GitEx$ cat a

This is the file a.

Fix!!!

果然，多了一个Fix!!!

不过要注意的是，如果master与Fix分支中间有多次提交，它会针对每次提交生成一个patch。

3.两种patch的比较：

•兼容性：很明显，git diff生成的Patch兼容性强。如果你在修改的代码的官方版本库不是Git管理的版本库，那么你必须使用git diff生成的patch才能让你的代码被项目的维护人接受。

•除错功能：对于git diff生成的patch，你可以用git apply --check 查看补丁是否能够干净顺利地应用到当前分支中；如果git format-patch 生成的补丁不能打到当前分支，git am会给出提示，并协助你完成打补丁工作，你也可以使用git am -3进行三方合并，详细的做法可以参考git手册或者《Progit》。从这一点上看，两者除错功能都很强。

•版本库信息：由于git format-patch生成的补丁中含有这个补丁开发者的名字，因此在应用补丁时，这个名字会被记录进版本库，显然，这样做是恰当的。因此，目前使用Git的开源社区往往建议大家使用format-patch生成补丁。