
This book is available in English.

Full translation available in

български език,
Español,
Français,
日本語,
한국어,
Nederlands,
Русский,
Українська
简体中文,

Partial translations available in

Čeština,
Indonesian,
Polski,

Српски,
Tagalog,
繁體中文,

Translations started for

Беларуская,
Deutsch,
فارسی,
Ελληνικά,
Italiano,
Bahasa Melayu,
Polski,
Português (Brasil),
Türkçe,
Ўзбекча.

The source of this book is hosted on GitHub.
Patches, suggestions and comments are welcome.

Chapters ▾ 1st Edition

.3 Git 工具 - 储藏(Stashing)

储藏(Stashing)

经常有这样的事情发生,当你正在进行项目中某一部分的工作,里面的东西处于一个比较杂乱的状态,而你想转到其他分支上进行一些工作。问题是,你不想提交进行了一半的工作,否则以后你无法回到这个工作点。解决这个问题的办法就是`git stash`命令。

“‘储藏”“可以获取你工作目录的中间状态——也就是你修改过的被追踪的文件和暂存的变更——并将它保存到一个未完结变更的堆栈中,随时可以重新应用。

储藏你的工作

为了演示这一功能,你可以进入你的项目,在一些文件上进行工作,有可能还暂存其中一个变更。如果你运行 `git status`,你可以看到你的中间状态:

```
$ git status
# On branch master
# Changes to be committed:
#   (use "git reset HEAD <file>..." to unstage)
#
#      modified:   index.html
#
# Changes not staged for commit:
#   (use "git add <file>..." to update what will be committed)
#
#      modified:   lib/simplegit.rb
#
```

现在你想切换分支,但是你还不想提交你正在进行中的工作;所以你储藏这些变更。为了往堆栈推送一个新的储藏,只要运行 `git stash`:

```
$ git stash
Saved working directory and index state \
  "WIP on master: 049d078 added the index file"
HEAD is now at 049d078 added the index file
(To restore them type "git stash apply")
```

你的工作目录就干净了:

```
$ git status
# On branch master
nothing to commit, working directory clean
```

这时,你可以方便地切换到其他分支工作;你的变更都保存在栈上。要查看现有的储藏,你可以使用 `git stash list`:

```
$ git stash list
stash@{0}: WIP on master: 049d078 added the index file
stash@{1}: WIP on master: c264051 Revert "added file_size"
stash@{2}: WIP on master: 21d80a5 added number to log
```

在这个案例中,之前已经进行了两次储藏,所以你可以访问到三个不同的储藏。你可以重新应用你刚刚实施的储藏,所采用的命令就是之前在原始的 stash 命令的帮助输出里提示的:`git stash apply`。如果你想应用更早的储藏,你可以通过名字指定它,像这样:`git stash apply stash@{2}`。如果你不指明,Git 默认使用最近的储藏并尝试应用它:

```
$ git stash apply
# On branch master
# Changes not staged for commit:
#   (use "git add <file>..." to update what will be committed)
#
#      modified:   index.html
#      modified:   lib/simplegit.rb
#
```

你可以看到 Git 重新修改了你所储藏的那些当时尚未提交的文件。在这个案例里,你尝试应用储藏的工作目录是干净的,并且属于同一分支;但是一个干净的工作目录和应用到相同的分支上

并不是应用储藏的必要条件。你可以在其中一个分支上保留一份储藏，随后切换到另外一个分支，再重新应用这些变更。在工作目录里包含已修改和未提交的文件时，你也可以应用储藏——Git 会给出归并冲突如果有任何变更无法干净地被应用。

对文件的变更被重新应用，但是被暂存的文件没有重新被暂存。想那样的话，你必须在运行 `git stash apply` 命令时带上一个 `--index` 的选项来告诉命令重新应用被暂存的变更。如果你是这么做的，你应该已经回到你原来的位置：

```
$ git stash apply --index
# On branch master
# Changes to be committed:
#   (use "git reset HEAD <file>..." to unstage)
#
#       modified:   index.html
#
# Changes not staged for commit:
#   (use "git add <file>..." to update what will be committed)
#
#       modified:   lib/simplegit.rb
#
```

apply 选项只尝试应用储藏的工作——储藏的内容仍然在栈上。要移除它，你可以运行 `git stash drop`，加上你希望移除的储藏的名字：

```
$ git stash list
stash@{0}: WIP on master: 049d078 added the index file
stash@{1}: WIP on master: c264051 Revert "added file_size"
stash@{2}: WIP on master: 21d80a5 added number to log
$ git stash drop stash@{0}
Dropped stash@{0} (364e91f3f268f0900bc3ee613f9f733e82aaed43)
```

你也可以运行 `git stash pop` 来重新应用储藏，同时立刻将其从堆栈中移走。

取消储藏(Un-applying a Stash)

在某些情况下，你可能想应用储藏的修改，在进行了一些其他的修改后，又要取消之前所应用储藏的修改。Git没有提供类似于 `stash unapply` 的命令，但是可以通过取消该储藏的补丁达到同样的效果：

```
$ git stash show -p stash@{0} | git apply -R
```

同样的，如果你沒有指定具体的某个储藏，Git 会选择最近的储藏：

```
$ git stash show -p | git apply -R
```

你可能会想要新建一个别名，在你的 Git 里增加一个 `stash-unapply` 命令，这样更有效率。例如：

```
$ git config --global alias.stash-unapply '!git stash show -p | git apply -R'
```

```
$ git stash apply
$ #... work work work
$ git stash-unapply
```

从储藏中创建分支

如果你储藏了一些工作，暂时不去理会，然后继续在你储藏工作的分支上工作，你在重新应用工作时可能会碰到一些问题。如果尝试应用的变更是针对一个你那之后修改过的文件，你会碰到一个归并冲突并且必须去化解它。如果你想用更方便的方法来重新检验你储藏的变更，你可以运行 `git stash branch`，这会创建一个新的分支，检出你储藏工作时的所处的提交，重新应用你的工作，如果成功，将会丢弃储藏。

```
$ git stash branch testchanges
Switched to a new branch "testchanges"
# On branch testchanges
# Changes to be committed:
#   (use "git reset HEAD <file>..." to unstage)
#
#       modified:   index.html
#
# Changes not staged for commit:
#   (use "git add <file>..." to update what will be committed)
#
#       modified:   lib/simplegit.rb
#
Dropped refs/stash@{0} (f0dfc4d5dc332d1cee34a634182e168c4efc3359)
```

这是一个很棒的捷径来恢复储藏的工作然后在新的分支上继续当时的工作。

prev | next
About this site
Patches, suggestions, and comments are welcome.
Git is a member of Software Freedom Conservancy