6.2 Git 工具 - 交互式暂存

交互式暂存

Git提供了很多脚本来辅助某些命令行任务。这里，你将看到一些交互式命令，它们帮助你方便地构建只包含特定组合和部分文件的提交。在你修改了一大批文件然后决定将这些变更分布在几个各有侧重的提交而不是单个又大又乱的提交时，这些工具非常有用。用这种方法，你可以确保你的提交在逻辑上划分为相应的变更集，以便于供和你一起工作的开发者审阅。如果你运行`git add`时加上`-i`或者`--interactive`选项，Git就进入了一个交互式的shell模式，显示一些类似于下面的信息：

```
$ git add -i
           staged     unstaged path
  1:    unchanged        +0/-1 TODO
  2:    unchanged        +1/-1 index.html
  3:    unchanged        +5/-1 lib/simplegit.rb

*** Commands ***
  1: status     2: update      3: revert     4: add untracked
  5: patch      6: diff        7: quit       8: help
What now>
```

你会看到这个命令以一个完全不同的视图显示了你的暂存区——主要是你通过`git status`得到的那些信息但是稍微简洁但信息更加丰富一些。它在左侧列出了你暂存的变更，在右侧列出了未被暂存的变更。

在这之后是一个命令区。这里你可以做很多事情，包括暂存文件，撤回文件，暂存部分文件，加入未被追踪的文件，查看暂存文件的差别。

暂存和撤回文件

如果你在`What now>`的提示后输入`2`或者`u`，这个脚本会提示你那些文件你想要暂存：

```
What now> 2
           staged     unstaged path
  1:    unchanged        +0/-1 TODO
  2:    unchanged        +1/-1 index.html
  3:    unchanged        +5/-1 lib/simplegit.rb
Update>>
```

如果想暂存TODO和index.html，你可以输入相应的编号：

```
Update>> 1,2
           staged     unstaged path
* 1:    unchanged        +0/-1 TODO
```

```
* 2:    unchanged        +1/-1 index.html
  3:    unchanged        +5/-1 lib/simplegit.rb
Update>>
```

每个文件旁边的`*`表示选中的文件将被暂存。如果你在`update>>`提示后直接敲入回车，Git会替你把所有选中的内容暂存：

```
Update>>
updated 2 paths

*** Commands ***
  1: status     2: update      3: revert     4: add untracked
  5: patch      6: diff        7: quit       8: help
What now> 1
           staged     unstaged path
  1:        +0/-1      nothing TODO
  2:        +1/-1      nothing index.html
  3:    unchanged        +5/-1 lib/simplegit.rb
```

现在你可以看到TODO和index.html文件被暂存了同时simplegit.rb文件仍然未被暂存。如果这时你想要撤回TODO文件，就使用`3`或者`r`（代表revert，恢复）选项：

```
*** Commands ***
  1: status     2: update      3: revert     4: add untracked
  5: patch      6: diff        7: quit       8: help
What now> 3
           staged     unstaged path
  1:        +0/-1      nothing TODO
  2:        +1/-1      nothing index.html
  3:    unchanged        +5/-1 lib/simplegit.rb
Revert>> 1
           staged     unstaged path
* 1:        +0/-1      nothing TODO
  2:        +1/-1      nothing index.html
  3:    unchanged        +5/-1 lib/simplegit.rb
Revert>> [enter]
reverted one path
```

再次查看Git的状态，你会看到你已经撤回了TODO文件

```
*** Commands ***
  1: status     2: update      3: revert     4: add untracked
  5: patch      6: diff        7: quit       8: help
What now> 1
```

```
           staged     unstaged path
  1:    unchanged        +0/-1 TODO
  2:        +1/-1      nothing index.html
  3:    unchanged        +5/-1 lib/simplegit.rb
```

要查看你暂存内容的差异，你可以使用`6`或者`d`（表示diff）命令。它会显示你暂存文件的列表，你可以选择其中的几个，显示其被暂存的差异。这跟你在命令行下指定`git diff --cached`非常相似：

```
*** Commands ***
  1: status     2: update      3: revert      4: add untracked
  5: patch      6: diff        7: quit        8: help
What now> 6
           staged     unstaged path
  1:        +1/-1      nothing index.html
Review diff>> 1
diff --git a/index.html b/index.html
index 4d07108..4335f49 100644
--- a/index.html
+++ b/index.html
@@ -16,7 +16,7 @@ Date Finder

 <p id="out">...</p>

-<div id="footer">contact : support@github.com</div>
+<div id="footer">contact : email.support@github.com</div>

 <script type="text/javascript">
```

通过这些基本命令，你可以使用交互式增加模式更加方便地处理暂存区。

暂存补丁

只让Git暂存文件的某些部分而忽略其他也是有可能的。例如，你对simplegit.rb文件作了两处修改但是只想暂存其中一个而忽略另一个，在Git中实现这一点非常容易。在交互式的提示符下，输入`5`或者`p`（表示patch，补丁）。Git会询问哪些文件你希望部分暂存；然后对于被选中文件的每一节，他会逐个显示文件的差异区块并询问你是否希望暂存他们：

```
diff --git a/lib/simplegit.rb b/lib/simplegit.rb
index dd5ecc4..57399e0 100644
--- a/lib/simplegit.rb
+++ b/lib/simplegit.rb
@@ -22,7 +22,7 @@ class SimpleGit
   end

   def log(treeish = 'master')
-    command("git log -n 25 #{treeish}")
```

```
+    command("git log -n 30 #{treeish}")
   end

   def blame(path)
Stage this hunk [y,n,a,d,/,j,J,g,e,?]?
```

此处你有很多选择。输入`?`可以显示列表：

```
Stage this hunk [y,n,a,d,/,j,J,g,e,?]? ?
y - stage this hunk
n - do not stage this hunk
a - stage this and all the remaining hunks in the file
d - do not stage this hunk nor any of the remaining hunks in the file
g - select a hunk to go to
/ - search for a hunk matching the given regex
j - leave this hunk undecided, see next undecided hunk
J - leave this hunk undecided, see next hunk
k - leave this hunk undecided, see previous undecided hunk
K - leave this hunk undecided, see previous hunk
s - split the current hunk into smaller hunks
e - manually edit the current hunk
? - print help
```

如果你想暂存各个区块，通常你会输入`y`或者`n`，但是暂存特定文件里的全部区块或者暂时跳过对一个区块的处理同样也很有用。如果你暂存了文件的一个部分而保留另外一个部分不被暂存，你的状态输出看起来会是这样：

```
What now> 1
           staged     unstaged path
  1:    unchanged        +0/-1 TODO
  2:        +1/-1      nothing index.html
  3:        +1/-1        +4/-0 lib/simplegit.rb
```

simplegit.rb的状态非常有意思。它显示有几行被暂存了，有几行没有。你部分地暂存了这个文件。在这时，你可以退出交互式脚本然后运行`git commit`来提交部分暂存的文件。

最后你也可以不通过交互式增加的模式来实现部分文件暂存——你可以在命令行下通过`git add -p`或者`git add --patch`来启动同样的脚本。