GIT REAL 2

STASHING

CHAPTER 2

STASHING

Gregg is halfway done with work on the "gerbils" branch, but an issue with "master" needs fixing NOW

```
$ git diff
diff --git a/index.html b/index.html
index d36fac4..d2923a8 100644
--- a/index.html
+++ b/index.html
@@ -7,6 +7,7 @@
   <body>
     <nav>
       <ul>
+         <li><a href="gerbil
          <li><a href="cat.html">Cats</a></li>
          <li><a href="dog.html">Dogs</a></li>
       </ul>
```

STASHING

```
$ git stash save
Saved working directory and index state
WIP on gerbils: b2bdead Add dogs.
```

saves modified files

restores last commit

```
$ git diff
$ git status
# On branch gerbils
nothing to commit (working directory clean)
```

Modifications are "stashed" away

STASHING

Gregg checks out "master", pulls changes, and makes fixes

```
$ git checkout master
Switched to branch 'master'
Your branch is behind 'origin/master' by 1 commit,
and can be fast-forwarded.

$ git pull
Updating b2bdead..686b55d
Fast-forward
 wolf.html | 7 +++++++
 1 file changed, 7 insertions(+)
 create mode 100755 wolf.html
```

LIST STASHES

Now he's ready to return to the feature branch

```
$ git checkout gerbils
Switched to branch 'gerbils'
```

APPLY STASHES

```
$ git stash apply
# On branch gerbils
# Changes not staged for commit:
#   (use "git add <file>..." to update what will be committed)
#   (use "git checkout -- <file>..." to discard changes)
#
#  modified:   index.html
```

Bring stashed files back

```
$ git diff
diff --git a/index.html b/index.html
       <ul>
+        <li><a href="gerbil
         <li><a href="cat.html">Cats</a></li>
         <li><a href="dog.html">Dogs</a></li>
       </ul>
```

Here are the changes!

APPLY STASHES

```
$ git stash list
stash@{0}: WIP on master: 686b55d Add wolves.
stash@{1}: WIP on gerbils: b2bdead Add dogs.
stash@{2}: WIP on gerbils: b2bdead Add dogs.
```

You can have multiple stashes

Stash names are shown in the list

```
$ git stash apply stash@{1}
# On branch gerbils
# Changes not staged for commit:
#
#   modified:   index.html
```

"stash@{0}" is the default when applying; specify the stash name to apply a different one

DROP STASHES

"git stash drop" discards a stash

```
$ git stash list
stash@{0}: WIP on gerbils: b2bdead Add dogs.
```

Stash has been applied but it's still here

```
$ git stash drop
Dropped (6dc716f...)
```

Delete it from list

```
$ git stash list
```

Old stash is gone!

SHORTCUTS

shortcut	same as
`git stash`	`git stash save`
`git stash apply`	`git stash apply stash@{0}`
`git stash drop`	`git stash drop stash@{0}`

`git stash pop`	`git stash apply`
	`git stash drop`

STASH CONFLICTS

```
$ git stash apply
error: Your local changes to the following files would be
overwritten by merge:
    index.html
Please, commit your changes or stash them before you can merge.
Aborting
```

Conflicts are possible when applying a stash

```
$ git reset --hard HEAD
```

Commit or reset your local changes, as appropriate

```
$ git stash apply
# On branch gerbils
# Changes not staged for commit:
...
#   modified:   index.html
```

success!

STASH CONFLICTS

```
$ git stash apply
Auto-merging index.html
CONFLICT (content): Merge conflict in index.html
```

You need to merge the conflicted lines as usual...

STASH CONFLICTS

```
$ git stash pop
Auto-merging index.html
CONFLICT (content): Merge conflict in index.html
```

If you're using the "pop" shortcut, and there's a conflict...

You *also* need to merge the conflicted lines as usual...

```
$ git stash list
stash@{0}: WIP on gerbils: b2bdead Add dogs.
```

And the stash won't be dropped automatically

```
$ git stash drop
Dropped refs/stash@{0} (e4ba8a4...)
```

So be sure to do it manually

KEEP INDEX

Jane wants to commit some changes saved in the staging area

```
$ git status
# On branch gerbils
# Changes to be committed:
#
#   new file:   gerbil.html
#
# Changes not staged for commit:
#
#   modified:   index.html
```

we want to commit this

and stash everything else

```
$ git stash save
Saved working directory and index state...

$ git status
# On branch gerbils
nothing to commit (working directory clean)
```

Whoops, she stashed everything

KEEP INDEX

Jane pops the stash to get her staged changes back

```
$ git stash pop
# On branch gerbils
# Changes to be committed:
#
#   new file:   gerbil.html
#
# Changes not staged for commit:
#
#   modified:   index.html
#
Dropped refs/stash@{0} (de4105a...)
```

Stashed staging areas get restored later

KEEP INDEX

```
$ git stash save --keep-index
Saved working directory and index state
HEAD is now at b2bdead Add dogs.
```

```
$ git status
# On branch gerbils
# Changes to be committed:
#
#   new file:   gerbil.html

$ git commit -m "Add gerbils section."
[gerbils 130a661] Add gerbils section.
 1 file changed, 7 insertions(+)
 create mode 100644 gerbil.html
```

"--keep-index" option causes the staging area not to be stashed

Now Jane can commit the changes

KEEP INDEX

```
$ git stash pop
# On branch gerbils
# Changes not staged for commit:
#
#   modified:    index.html
#
no changes added to commit
Dropped refs/stash@{0} (db990c0...)
```

Unstaged changes get stashed and restored as usual

INCLUDE UNTRACKED

Jane needs to stash changes, but she has an untracked file

```
$ git stash save
Saved working directory and index state
HEAD is now at b2bdead Add dogs.

$ git status
# On branch gerbils
# Untracked files:
#
#   gerbil.html
```

Normally, only tracked files get stashed

INCLUDE UNTRACKED

```
$ git stash save --include-untracked
Saved working directory and index state
HEAD is now at b2bdead Add dogs.
```

```
$ git status
# On branch gerbils
nothing to commit (working directory clean)

$ git stash pop
# On branch gerbils
# Changes not staged for commit:
#
#   modified:   index.html
#
# Untracked files:
#
#   gerbil.html
```

"--include-untracked" option causes untracked files to be stashed, too

When stash is restored, untracked files will be, too

LIST OPTIONS

```
$ git stash list
stash@{0}: WIP on gerbils: b2bdead Add dogs.
stash@{1}: WIP on master: 686b55d Add wolves.
stash@{2}: WIP on gerbils: b2bdead Add dogs.
```

```
$ git stash list --stat
stash@{0}: WIP on gerbils: b2bdead Add dogs.
 gerbil.html | 7 +++++++
 index.html  | 1 +
 wolf.html   | 7 -------
 3 files changed, 8 insertions(+), 7 deletions(-)

stash@{1}: WIP on master: 686b55d Add wolves.
 gerbil.html | 7 -------
 ...
```

When you have more than one stash, it's hard to tell them apart

"git stash list" can take any option "git log" can
For example, "--stat" summarizes file changes

STASH SHOW

```
$ git stash show stash@{0}
 gerbil.html | 7 +++++++
 index.html  | 1 +
 2 files changed, 8 insertions(+)
```

Shows one particular stash

```
$ git stash show
 gerbil.html | 7 +++++++
 index.html  | 1 +
 2 files changed, 8 insertions(+)
```

Like "apply" and "drop", acts on most recent stash by default

STASH SHOW

```
$ git stash show --patch
diff --git a/gerbil.html b/gerbil.html
+<!DOCTYPE html>
+<html lang="en">
+  <head>
+    <meta charset="UTF-8">
+    <title>Our Gerbils</title>
+  </head>
+</html>
diff --git a/index.html b/index.html
   <body>
     <nav>
       <ul>
+        <li><a href="gerbil
         <li><a href="cat.html">Cats</a></li>
         <li><a href="dog.html">Dogs</a></li>
       </ul>
```

Also takes any option "git log" can

For example, "--patch" shows file diffs

STASH MESSAGES

```
$ git stash save "Add gerbils page, start index."
Saved working directory and index state
HEAD is now at b2bdead Add dogs.
```

You can provide a stash message when saving

```
$ git stash list
stash@{0}: On gerbils: Add gerbils page, start index.
stash@{1}: WIP on gerbils: b2bdead Add dogs.
stash@{2}: WIP on master: 686b55d Add wolves.
stash@{3}: WIP on gerbils: b2bdead Add dogs.
```

Gets listed in place of default message

BRANCHING

Gregg has added a gerbils section and is working on the toys page, when management asks him to deploy the branch ASAP

```
$ git stash save "Start toy section."
Saved working directory and index state
HEAD is now at 5e3bde9 Add gerbils page.
```

He stashes his changes...

Gregg accidentally deletes this branch, out of habit

```
$ git branch -d gerbils
Deleted branch gerbils (was 5e3bde9).
```

BRANCHING

Now, he needs a new branch to restore the stashed toys page

new branch name

stash to pop

```
$ git stash branch gerbil-toys stash@{0}
Switched to a new branch 'gerbil-toys'
# On branch gerbil-toys
# Changes not staged for commit:
#
#   modified:   gerbil.html
#
Dropped stash@{0} (5797b65...)
```

"git stash branch" checks a new branch out automatically...

and drops the stash automatically

```
$ git commit -am "Add gerbil toys."
[gerbil-toys e7083eb] Add gerbil toys.
 1 file changed, 3 insertions(+)
```

New branch is an ordinary branch, ready for commits

CLEAR STASHES

We have a lot of stashes we no longer need

```
$ git stash list
stash@{0}: WIP on gerbils: b2bdead Add dogs.
stash@{1}: WIP on master: 686b55d Add wolves.
stash@{2}: WIP on gerbils: b2bdead Add dogs.
```

```
$ git stash clear
```

Clears all of them at once

```
$ git stash list
```

All gone!