9. CHANGES, NOT FILES

Goals

- Understanding that git works with the changes, not the files.

Most version control systems work with files. You add the file to source control and the system tracks changes from that moment on.

Git concentrates on the changes to a file, not the file itself. A `git add file` command does not tell git to add the file to the repository, but to note the current state of the file for it to be commited later.

We will try to investigate the difference in this lesson.

First Change: Adding default page tags

Change the "Hello, World" page so that it contained default tags `<html>` and `<body>`.

FILE: *HELLO.HTML*

```
<html>
  <body>
    <h1>Hello, World!</h1>
  </body>
</html>
```

Add this change

Now add this change to the git staging.

RUN:

```
git add hello.html
```

Second change: Add the HTML headers

Now add the HTML headers (`<head>` section) to the "Hello, World" page.

FILE: *HELLO.HTML*

```
<html>
  <head>
  </head>
  <body>
    <h1>Hello, World!</h1>
  </body>
</html>
```

Check the current status

RUN:

```
git status
```

You will see ...

RESULT:

```
$ git status
# On branch master
# Changes to be committed:
#   (use "git reset HEAD <file>..." to unstage)
#
#       modified:   hello.html
#
# 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:   hello.html
#
```

Please note that hello.html is listed in the status twice. The first change (the addition of default tags) is istaged and ready for a commit. The second change (adding HTML headers) is unstaged. If you were making a commit right now, headers would not have been saved to the repository.

Let's check.

Commit

Commit the staged changes (default values), then check the status one more time.

RUN:

```
git commit -m "Added standard HTML page tags"
git status
```

You will see ...

RESULT:

```
$ git commit -m "Added standard HTML page tags"
[master 8c32287] Added standard HTML page tags
 1 files changed, 3 insertions(+), 1 deletions(-)
$ git status
# On branch master
# 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:   hello.html
#
no changes added to commit (use "git add" and/or "git commit -a")
```

The status command suggests that `hello.html` has unrecorded changes, but is no longer in the buffer zone.

Adding the second change

Add the second change to the staging area, after that run the `git status` command.

RUN:

```
git add .
git status
```

Note: The current directory ('.') will be our file to add. This is the most convenient way to add all the changes to the files of the current directory and its folders. But since it adds everything, it is a good idea to check the status prior to doing an `add .`, to make sure you don't add any file that should not be added.

I wanted you to see the "add ." trick, and we will continue adding explicit files later on just in case.

You will see ...

RESULT:

```
$ git status
# On branch master
# Changes to be committed:
#   (use "git reset HEAD <file>..." to unstage)
#
#       modified:   hello.html
#
```

The second change has been staged and is ready for a commit.

Commit the second change

RUN:

```
git commit -m "Added HTML header"
```

‹ 8. Commiting the changes

10. History ›