**Exercise - Use Git to fix mistakes**

* 15 minutes

Now, let's get some hands-on practice fixing our mistakes!

**Practice recovering a deleted file**

1. First, try deleting *index.html*:

Bash

rm index.html

It might seem like a bad idea, but remember: Git has your back!

1. Use an ls command to verify that *index.html* was deleted:

Bash

ls

1. You should see the following output. Note that now, there's no *index.html* file!

Output

CSS

1. Let's recover *index.html*. Use git checkout to bring back index.html:

Bash

git checkout -- index.html

1. Use ls again to check the contents of the current directory. Has *index.html* been restored?

Yes! Now, the output should have an *index.html* file and a *CSS* directory:

Output

CSS index.html

**Practice recovering a file that was deleted: git rm**

When you want to recover deleted files, things are a little more complicated if you delete them by using git rm instead of by using rm.

1. To see what happens, try this command:

Bash

git rm index.html

1. Again, look for *index.html* by running ls. You should not see *index.html*.
2. Try to recover *index.html* the same way that you did last time:

Bash

git checkout -- index.html

1. This time, Git complains that it knows nothing about *index.html*. That's because Git not only deleted the file, it recorded the deletion in the index:

Output

error: pathspec 'index.html' did not match any file(s) known to git.

1. Unstage the deletion of *index.html* with the git reset command:

Bash

git reset HEAD index.html

1. Check for this output, which confirms it:

Output

Unstaged changes after reset:

D index.html

1. Now, you can recover your file from the index with the command you used before:

Bash

git checkout -- index.html

git reset unstaged the change, but the file was still deleted, so you had to use checkout to get it back.

1. Double-check that it worked by running ls.

**Revert a commit**

Now, let's make things more complicated. Suppose you accidentally overwrite one file with another file, or you make a change to a file that turns out to be a big mistake. You want to revert to the previous version of the file, but you had already committed the changes. In this case, a simple git checkout won't do the trick.

One solution to this problem is to revert the previous commit.

1. Open *index.html* with code:

Bash

code index.html

1. Replace the contents of *index.html* with this code:

HTML

<h1>That was a mistake!</h1>

1. Save and close the file.
2. Use these commands to commit the changes, and show the latest commit:

Bash

git commit -m "Purposely overwrite the contents of index.html" index.html

git log -n1

The -n1 flag here tells Git that we want only the most recent commit entry.

1. Use the following commands to attempt to restore *index.html*:

Bash

git checkout -- index.html

1. Open *index.html* in the editor:

Bash

code index.html

Which version of *index.html* do you see? The old version or the new version?

In this situation, the best course of action is to *revert* the change by making another commit that cancels out the first one. That's a job for git revert.

1. Close the file and use git revert to undo your committed changes:

Bash

git revert --no-edit HEAD

The --no-edit flag tells Git that we don't want to add a commit message for this action.

1. Check the output. It should look similar to this example:

Output

[main 6a27310] Revert "Purposely overwrite the contents of index.html"

1 file changed, 13 insertions(+), 1 deletion(-)

1. Follow up with a git log command to show the latest commit:

Bash

git log -n1

1. Check the output again. It should look like this example:

Output

Author: User Name <user-name@contoso.com>

Date: Tue Nov 19 23:42:26 2019 +0000

Revert "Purposely overwrite the contents of index.html"

This reverts commit 15d3bded388470c98881a632025bc15190fe9d17.

1. Finally, open the *index.html* file to make sure the content is the correct version.

Reverting isn't the only way to remedy this situation; you could just edit *index.html* and commit the corrected file. That option is harder if the changes you committed were extensive. In any case, git revert is a good way to signal your intent.