(/learn)

Lose a Commit, Get it Back

Learn how you can remove a commit and then retrieve it.

We'll cover the following

- Add commits
- Remove commit
- Retrieve commit
    - Use git reset to restore state

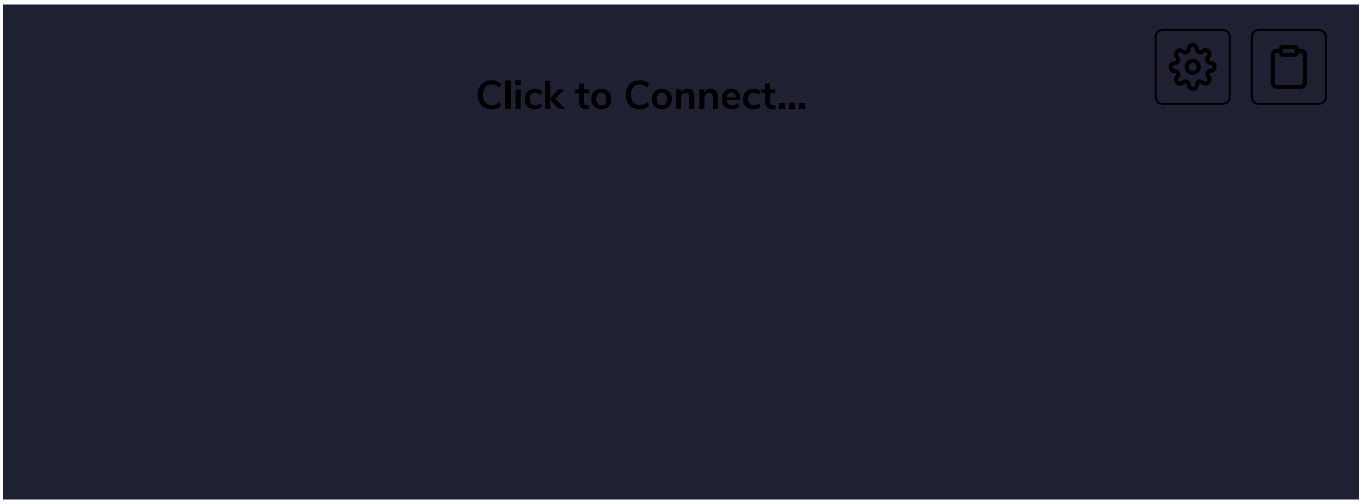
Add commits

First, set up a repository with two commits:

```
mkdir lgthw_reflog
cd lgthw_reflog
git init
echo first commit > file1
git add file1
git commit -m file1
echo second commit >> file1
git commit -am 'commit message for file1.1'
git log
```

Terminal 1

Terminal



Remove commit

Then do some magic to effectively remove the last commit by entering the following commands in the terminal given above:

```
git checkout HEAD^
git branch -f master
git checkout master
git log
```

What was that?
Don’t worry about what you just did; it’s a more advanced set of commands that mess with Git’s history. We’ll cover it later.

The last commit has disappeared! You have fully reverted the `master` branch to where it was before. Even `git log --all` does not show it because it’s not on a branch.

Don’t worry about the details of what you did! The point here is to create a situation in Git that you want to get out of.

Retrieve commit

This is where Git’s reflog can help.

Git reflog records all *movements* of branches in the repository. As with `git stash`es, it is local to your repository.

```
git reflog
```

Terminal 1

Terminal

> **Reflog?**
>
> The reflog is called that because it's a "REFerence LOG". I always think of someone being flogged, but that's probably memories of painful Git experiences...

Git's reflog is a history of the changes made to the `HEAD` (remember the `HEAD` is a pointer to the current location of the repository).

Use `git reset` to restore state

If you `git reset --hard` the repository to the given reference (in this case, `40e99f7`; your ID will differ!):

```
git reset --hard 40e99f7
git log
```

Terminal 1

Terminal

You are returned to where you were.

The `--hard` flag updates both the index (staging/added) and the working

The --hard flag updates both the index (staging/added) and the working tree as you saw previously.

The reflog contains references to the state of the repository at various points, *even if those points are no longer apparently reachable within the repository*.

← Back

Introduction: Reflog

Next →

Challenge: Reflog

☑ Mark as Completed

Report an Issue

Ask a Question (https://discuss.educative.io/tag/lose-a-commit-get-it-back__reflog__learn-git-the-hard-way)