GIT

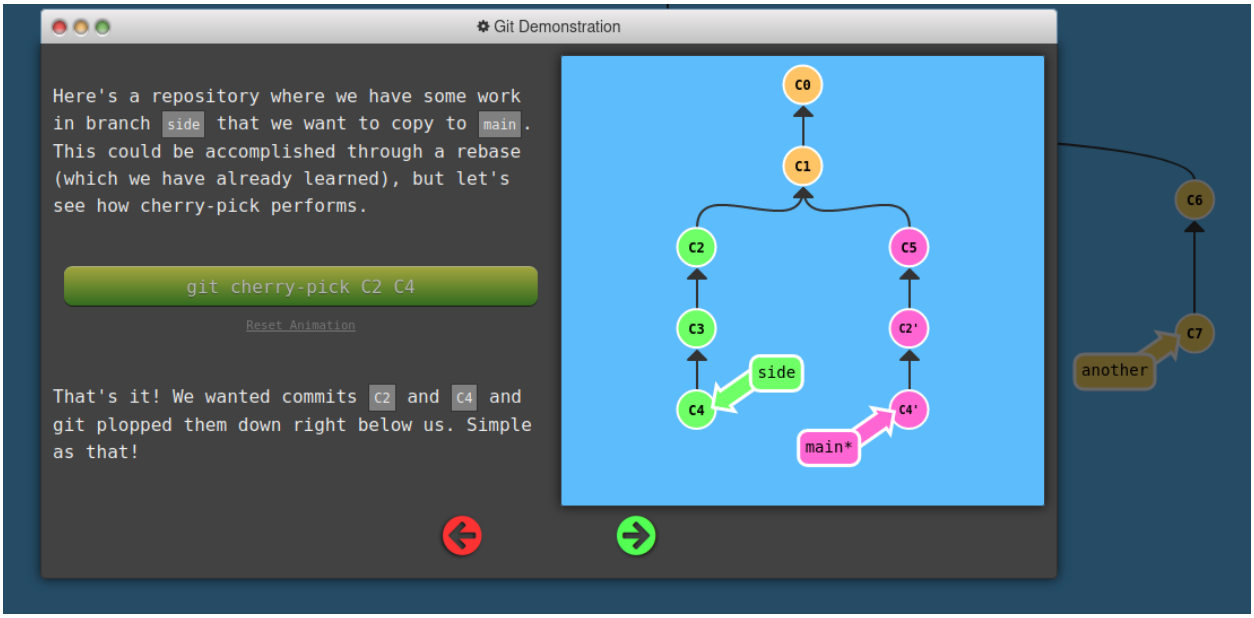
Level 3

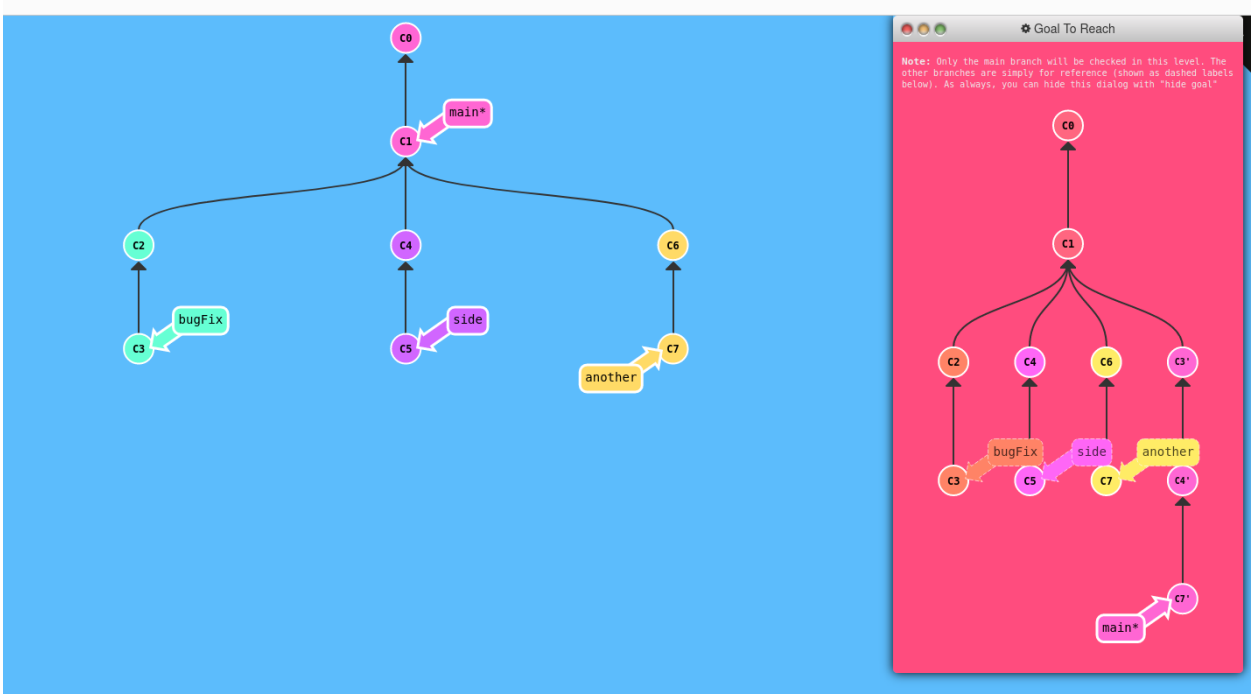
Task 1

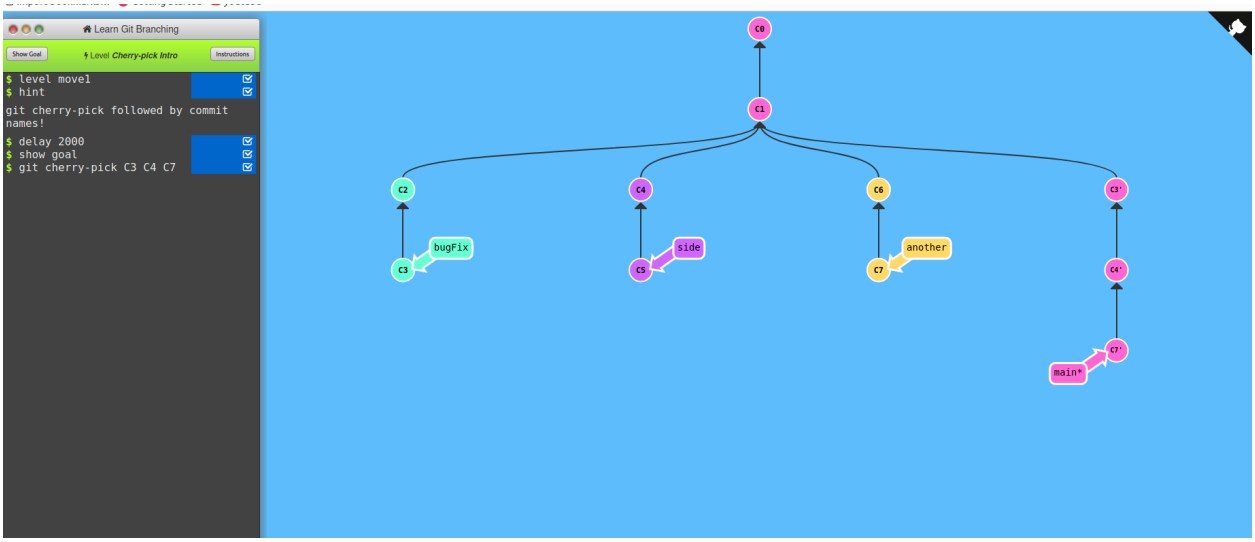
The first command in this series is called `git cherry-pick`. It takes on the following form:

- `git cherry-pick <Commit1> <Commit2> <...>`

It's a very straightforward way of saying that you would like to copy a series of commits below your current location (`HEAD`).








```
git checkout C3 C4 C7
```

Task 2

Git Interactive Rebase

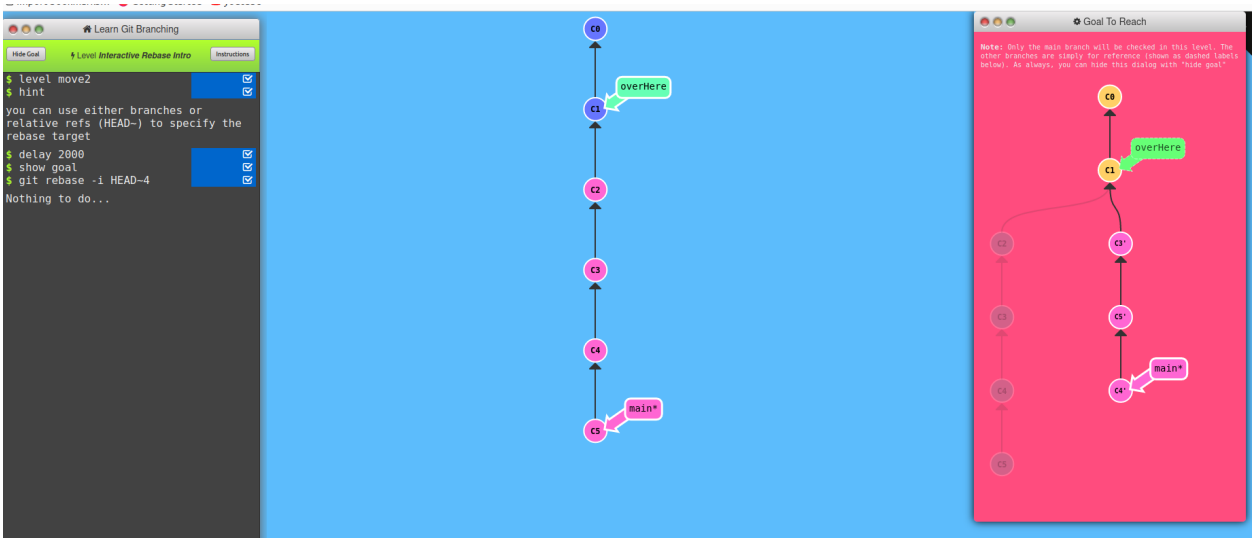
Git cherry-pick is great when you know which commits you want (*and* you know their corresponding hashes) -- it's hard to beat the simplicity it provides.

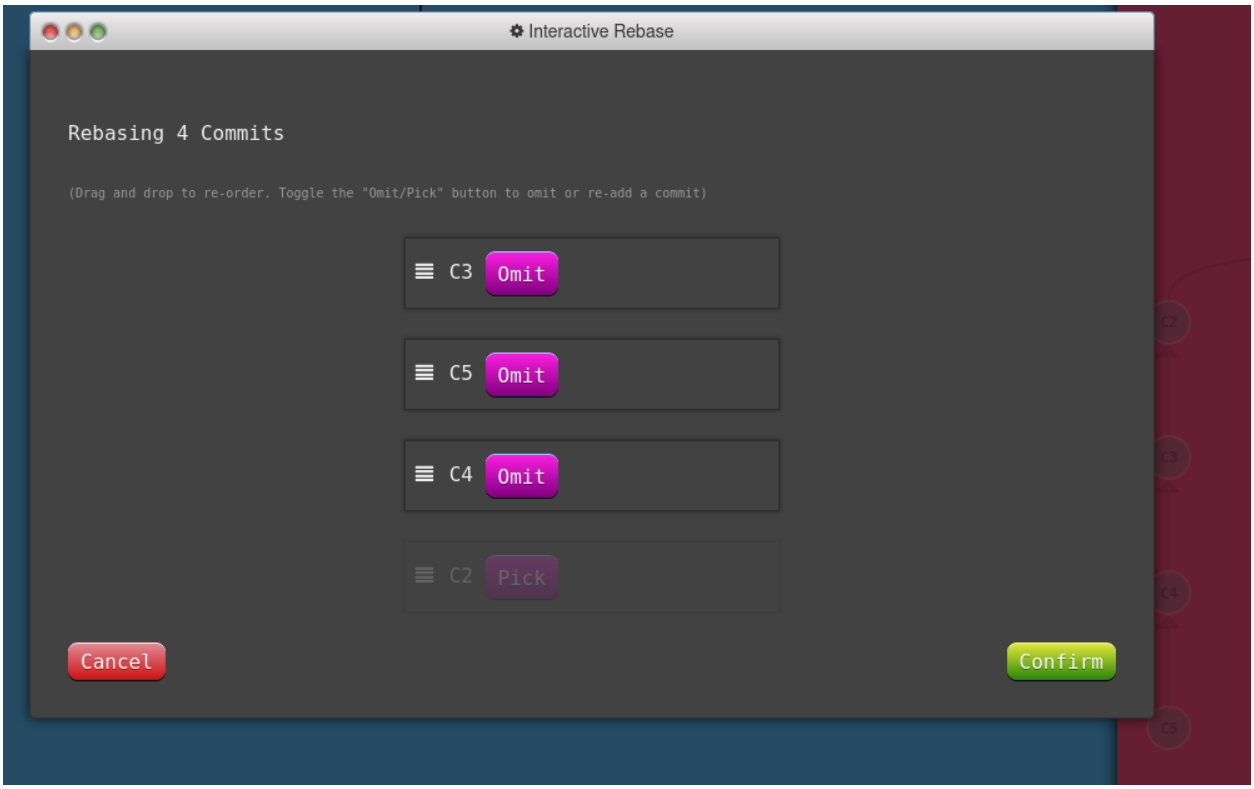
But what about the situation where you don't know what commits you want? Thankfully git has you covered there as well! We can use interactive rebasing for this -- it's the best way to review a series of commits you're about to rebase.

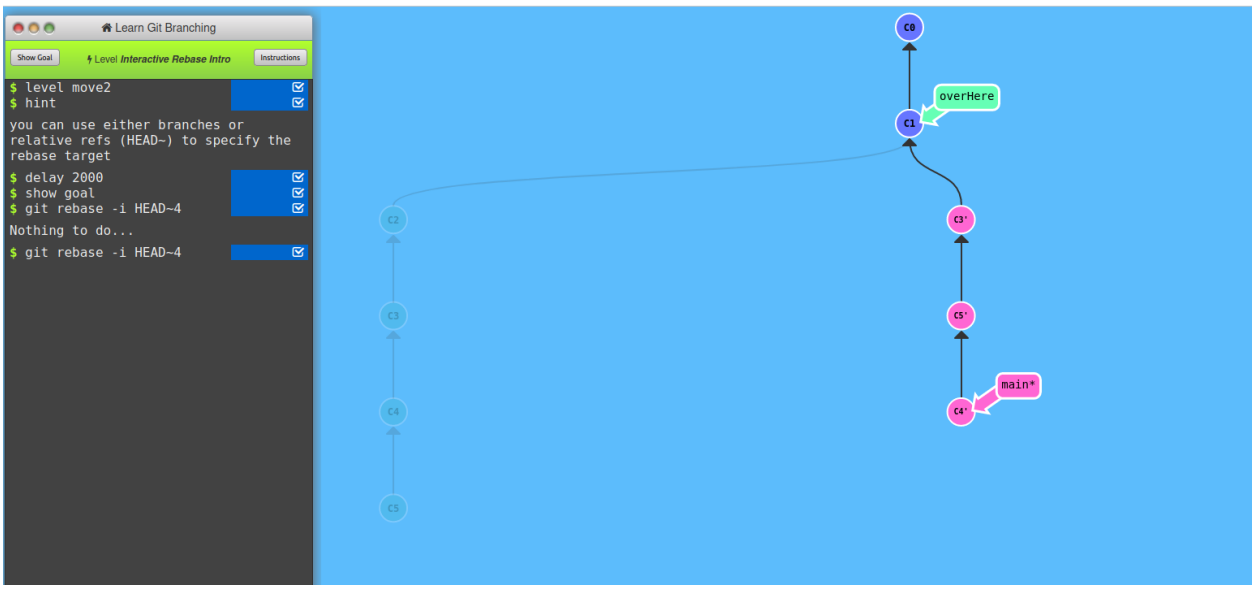
All interactive rebase means Git is using the `rebase` command with the `-i` option.

If you include this option, git will open up a UI to show you which commits are about to be copied below the target of the rebase. It also shows their commit hashes and messages, which is great for getting a bearing on what's what.

For "real" git, the UI window means opening up a file in a text editor like `vim`. For our purposes, I've built a small dialog window that behaves the same way.








```
git rebase -i HEAD~4
```

From above UI we omit the commit and arrange the order of the commit according to challenged specified.