30538 Problem Set 3: git Solution

Anand Kshirsagar

2024-10-21

SOLO

Learn git branching (15 points)

Go to https://learngitbranching.js.org. This is the best visual git explainer we know of.

1. Complete all the levels of main "Introduction Sequence". Report the commands needed to complete "Git rebase" with one line per command.

```
git check out  -b bugfix
git commit
git checkout main
git commit
git checkout bugfix
git rebase main
```

2. Complete all the levels of main "Ramping up". Report the commands needed to complete "Reversing changes in git" with one line per command.

```
git reset HEAD-1
git checkout pushed
git revert HEAD
```

3. Complete all the levels of remote "Push & Pull – Git Remotes!". Report the commands needed to complete "Locked Main" with one line per command.

```
git reset --hard o/main
git checkout -b feature C2
git push origin feature
```

Exercises

- Basic Staging and Branching (10-15)

1. Exercise. For your pset submission, tell us only the answer to the last question (22).

```
On branch master
nothing to commit, working tree clean
```

2. Exercise. For your pset submission, tell us only the output to the last question (18).

```
diff --git a/file1.txt b/file1.txt
deleted file mode 100644
index 4d835eb..0000000
--- a/file1.txt
+++ /dev/null
@@ -1 +0,0 @@
-Anand Kshirsagar
diff --git a/file2.txt b/file2.txt
new file mode 100644
index 0000000..030db67
--- /dev/null
+++ b/file2.txt
@@ -0,0 +1 @@
+This is file2.txt
```

- Merging

1. Exercise. After completing all the steps (1 through 12), run git log –oneline –graph –all and report the output.

```
* ac2030f (HEAD -> master, feature/uppercase) Update greeting.txt to contain
an uppercase greeting
* d61de5d Add content to greeting.txt
* fab9cde Add file greeting.txt
```

2. Exercise. Report the answer to step 11.

```
*   <merge_commit_hash> (HEAD -> master) Merge branch 'greeting'
|\
| * <greeting_commit_hash> Update greeting.txt with my favorite greeting
* | <readme_commit_hash> Add README.md with repository information
* | <previous_commit_hash> <previous_commit_message>
|/
* <initial_commit_hash> <initial_commit_message>
```

3. Identify the type of merge used in Q1 and Q2 of this exercise. In words, explain the difference between the two merge types, and describe scenarios where each type would be most appropriate.

ANSWER - Q1 - Fast-Forward Merge: This occurs when the feature/uppercase branch is merged into master with no new commits on master since the branch's creation.

Q2 - Three-Way Merge: This happens when merging the greeting branch into master after both branches have had changes.

Key Differences:

Fast-Forward Merge: Keeps a straight history without merge commits; best for feature branches without conflicting changes. Three-Way Merge: Creates a merge commit for more complex histories, used when both branches have diverged and need conflict resolution. Understanding these merges helps teams manage Git workflows and maintain a clear project history.

- Undo, Clean, and Ignore

1. Exercise. Report the answer to step 13.

```
fatal: ambiguous argument 'abc1234': unknown revision or path not in the
working tree.
Use '--' to separate paths from revisions, like this:
'git <command> [<revision>...]-- [<file>...]'
```

2. Exercise. Look up `git clean` since we haven't seen this before. For context, this example is about cleaning up compiled C code, but the same set of issues apply to random files generated by knitting a document or by compiling in Python. Report the terminal output from step 7.

```
$ ls
greeting.txt  README.md  other_file.txt
 Removing README.txt~
 Removing obj/
 Removing src/myapp.c~
 Removing src/oldfile.c~
```

3. Exercise. Report the answer to 15 ("What does git status say?")

```
On branch master
Changes to be committed:
(use "git restore--staged <file>..." to unstage)
deleted:
```

```
file1.txt
modified: .gitignore
new file: file3.txt
```