Lecture 8 - When Things Go Wrong

DSE 511

Drew Schmidt
2022-09-20

Announcements

- Nothing unresolved from last time
- Homework soon
- Questions?

Content

When Things Go Right

Working Locally

- `git init` -- Initialize a new repo
- `git status` -- Summarize the current state
- `git add` -- Stage a file for tracking
- `git commit` -- Commit staged changes

Working with Remotes

- `git clone` -- Download repo
- `git push` -- Send changes to remote
- `git pull` -- Get changes from remote

Starting Over the Demo Repo

```
git init
```

```
Initialized empty Git repository in /tmp/demo/.git/
```

```
git st
```

```
On branch master

No commits yet

nothing to commit (create/copy files and use "git add" to track)
```

Staging and File Editing

Some Common Problems with Staging and File Editing

- What if you stage the wrong file?
  - Maybe you changed your mind
  - Maybe you accidentally did `git add .`
- What if you want to throw away your changes?
  - Maybe it just doesn't work and you want to start over

Unstaging

```
touch x.txt
git add x.txt
git status
```

```
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
	new file:   x.txt
```

Unstaging

```
git reset x.txt
git status
```

```
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    x.txt

nothing added to commit but untracked files present (use "git add" to track)
```

Throwing Away Changes

```
touch y.txt
git add y.txt
git commit -m "add y.txt"
```

```
[master (root-commit) fe188e9] add y.txt
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 y.txt
```

Throwing Away Changes

```
echo "changes" >> y.txt
git status
```

```
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   y.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    x.txt

no changes added to commit (use "git add" and/or "git commit -a")
```

Throwing Away Changes

```
git restore y.txt
git status
```

```
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    x.txt

nothing added to commit but untracked files present (use "git add" to track)
```

```
git clean -f
```

```
Removing x.txt
```

```
git status
```

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

Editing Commits

Editing Commit

- What if you wrote the wrong thing?
- What if you added the wrong thing(s)?

Important caveat: we're assuming you haven't pushed yet!

Amend

- `git commit --amend` -- ADD to commit
- `git reset --soft HEAD~1` -- restore previous commit to staging

Editing a Commit Message

```
git status
```

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

```
git log
```

```
commit fe188e9e79f780f5b5ce84e8e980cdedaef0b7da (HEAD -> master)
Author: Drew Schmidt <wrathematics@gmail.com>
Date:   Mon Sep 19 17:42:21 2022 -0400

    add y.txt
```

Editing a Commit Message

```
git commit --amend -m "Add y.txt"
```

```
[master e9ec342] Add y.txt
 Date: Mon Sep 19 17:42:21 2022 -0400
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 y.txt
```

```
git log
```

```
commit e9ec3428b0e8ea8bd9a5ef39838c1bbdda505e54 (HEAD -> master)
Author: Drew Schmidt <wrathematics@gmail.com>
Date:   Mon Sep 19 17:42:21 2022 -0400

    Add y.txt
```

Amending a Commit

```
touch x.txt
git status
```

```
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    x.txt

nothing added to commit but untracked files present (use "git add" to track)
```

Amending a Commit

```
git add x.txt
git status
```

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

Amending a Commit

```
git commit --amend
```

```
[master dc75189] Add y.txt
 Date: Mon Sep 19 17:42:21 2022 -0400
 2 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 x.txt
 create mode 100644 y.txt
```

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

Amending a Commit

```
git commit --amend -m "Add x.txt and y.txt"
```

```
[master 7f0dca2] Add x.txt and y.txt
 Date: Mon Sep 19 17:42:21 2022 -0400
 2 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 x.txt
 create mode 100644 y.txt
```

Amending a Commit

```
git status
```

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

Amending a Commit

```
git log
```

```
commit 7f0dca293f49b3f20eb56255d127b3ace8f5371e (HEAD -> master)
Author: Drew Schmidt <wrathematics@gmail.com>
Date:   Mon Sep 19 17:42:21 2022 -0400

    Add x.txt and y.txt
```

```
ls
```

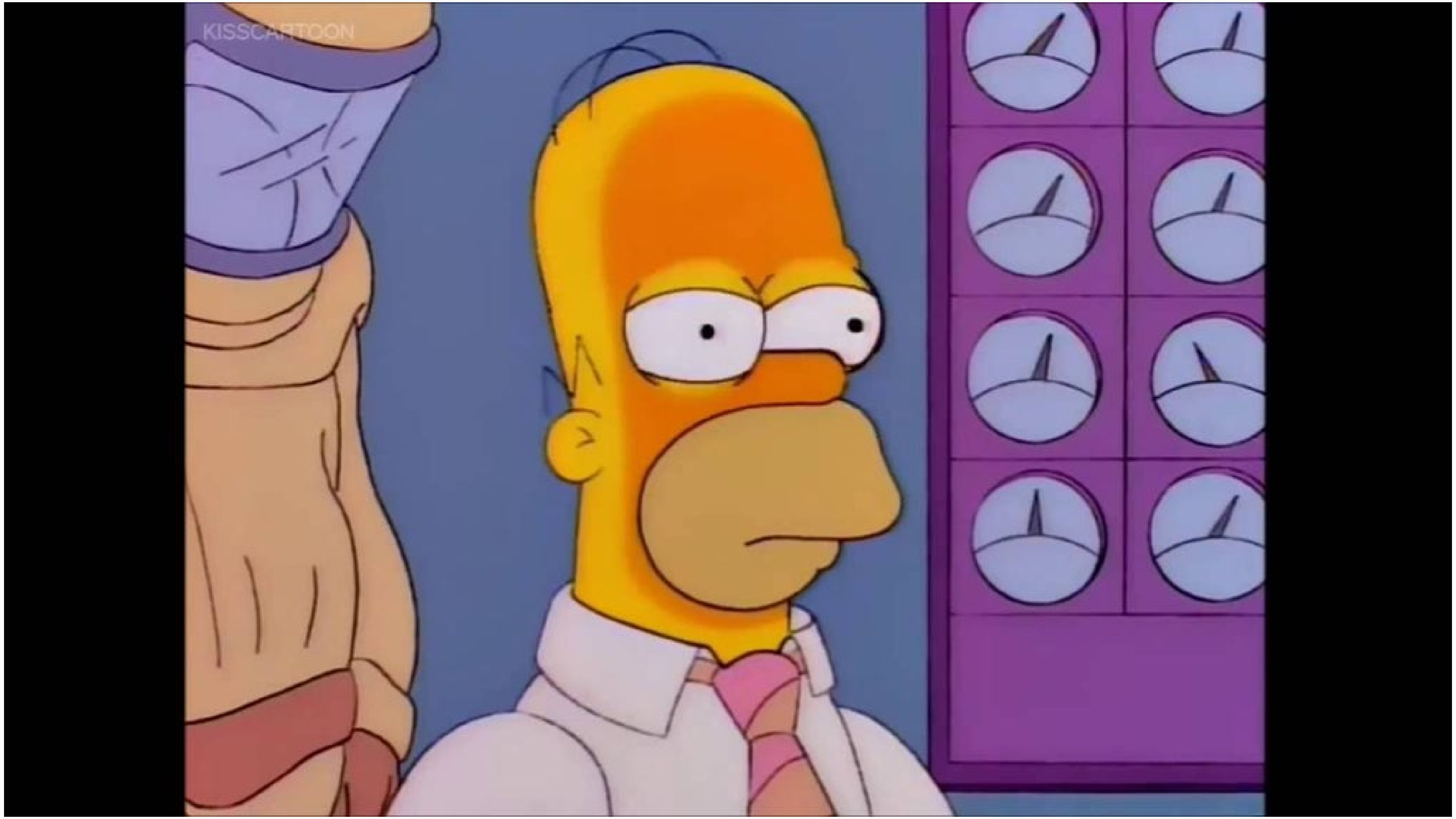
```
x.txt  y.txt
```

Amending a Commit

```
git reset --soft HEAD~1
```

```
fatal: ambiguous argument 'HEAD~1': unknown revision or path not in the working tree.
Use '--' to separate paths from revisions, like this:
'git <command> [<revision>...] -- [<file>...]'
```

We Have to Treat the First Commit Differently



Amending a Commit

```
touch z.txt
git add z.txt
git commit -m "add z.txt"
```

```
[master fc47cb9] add z.txt
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 z.txt
```

```
git reset --soft HEAD~1
git status
```

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

Amending a Commit

```
git restore --staged z.txt
rm z.txt
git status
```

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

```
git log
```

```
commit f0cb63163ca3be6ffbf6b56ac64aa2a3d5b8f790 (HEAD -> master)
Author: Drew Schmidt <wrathematics@gmail.com>
Date:   Mon Sep 19 17:42:21 2022 -0400

    Add x.txt and y.txt
```

Amending a Commit

```
git update-ref -d HEAD
git status
```

```
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   x.txt
    new file:   y.txt
```

```
git restore --staged x.txt y.txt
```

```
fatal: could not resolve HEAD
```

Amending a Commit

```
git rm --cached x.txt y.txt
```

```
rm 'x.txt'
rm 'y.txt'
```

```
git status
```

```
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    x.txt
    y.txt

nothing added to commit but untracked files present (use "git add" to track)
```

Amending a Commit

```
rm y.txt
git add x.txt
git commit -m "Add x.txt"
```

```
[master (root-commit) a7de297] Add x.txt
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 x.txt
```

Amending a Commit

```
git status
```

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

```
git log
```

```
commit a7de29721417d929841e8790bdca7723236f84f2 (HEAD -> master)
Author: Drew Schmidt <wrathematics@gmail.com>
Date:   Mon Sep 19 18:14:42 2022 -0400

    Add x.txt
```

Disagreements with Remotes

Disagreements with Remotes

- What if the local has "the truth"?
- What if the remote has "the truth"?
- ~~What if both have "the truth"?~~
  - AKA we want to merge
  - Handled separately

Making Remote Agree with Local

Scenario:

- We add things we didn't mean to (see last section)
- We accidentally pushed!
- We undo locally (as in last section) …
- Now what?

Making Remote Agree with Local

```
git push -f origin master
```

Be careful with this!

Making Local Agree with Remote

Scenario:

- We're working off of master instead of another branch
- We added some things
- It's not really working
- How do we abandon this?

We've Got Some Options

- Have we not been making commits?
  - `git reset`
- Have we been making commits?
  - Purge commits as in previous section
  - Is there a "force pull"?

Force Pull

```
git fetch --all
git reset --hard origin/master
```

Merges

Merging

- We saw a basic merge in Lecture 5
- It had no conflicts
- Dealing with conflicts can be tricky
- Fortunately the git isn't that bad

Merging

```
git status
```

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

```
git log
```

```
commit 89abacd0a6024d40cf881d87876683cfe898edea (HEAD -> master)
Author: Drew Schmidt <wrathematics@gmail.com>
Date:   Tue Sep 17 13:08:19 2022 -0400

    Add readme
```

Merging

```
ls
```

```
README.md
```

```
cat README.md
```

```
This is an example readme.
```

Merging

```
git checkout -b update-readme
```

```
Switched to a new branch 'update-readme'
```

```
vim README.md
cat README.md
```

```
This is an example readme.

It will eventually contain more information.
```

Merging

```
git diff
```

```
diff --git a/README.md b/README.md
index 28d1685..ae19468 100644
--- a/README.md
+++ b/README.md
@@ -1 +1,3 @@
 This is an example readme.
+
+It will eventually contain more information.
```

```
git add README.md
git commit -m "fill out readme"
```

```
[update-readme 1333b20] fill out readme
 1 file changed, 2 insertions(+)
```

Merging

```
git checkout master
```

```
Switched to branch 'master'
```

```
vim README.md
git diff README.md
```

```
diff --git a/README.md b/README.md
index 28d1685..635cc2c 100644
--- a/README.md
+++ b/README.md
@@ -1 +1,3 @@
 This is an example readme.
+
+This repository will be used to demonstrate a merge conflict.
```

Merging

```
git add README.md
git commit -m "add merge conflict note to readme"
```

```
[master b9c0773] add merge conflict note to readme
 1 file changed, 2 insertions(+)
```

```
git status
```

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

```
git branch
```

```
* master
  update-readme
```

Merging

```
git merge update-readme
```

```
Auto-merging README.md
CONFLICT (content): Merge conflict in README.md
Automatic merge failed; fix conflicts and then commit the result.
```

```
git status
```

```
On branch master
You have unmerged paths.
  (fix conflicts and run "git commit")
  (use "git merge --abort" to abort the merge)

Unmerged paths:
  (use "git add <file>..." to mark resolution)
    both modified:   README.md

no changes added to commit (use "git add" and/or "git commit -a")
```

Merging

```
cat README.md
```

```
This is an example readme.

<<<<<<< HEAD
This repository will be used to demonstrate a merge conflict.
=======
It will eventually contain more information.
>>>>>>> update-readme
```

Merging

```
git diff
```

```
diff --cc README.md
index 635cc2c,ae19468..0000000
--- a/README.md
+++ b/README.md
@@@ -1,3 -1,3 +1,3 @@@
  This is an example readme.

- This repository will be used to demonstrate a merge conflict.
 -It will eventually contain more information.
++This repository will be used to demonstrate a merge conflict. It will eventually contain more info
```

Merging

```bash
git add README.md
git commit -m "merge"
```

~~~
[master 80e0fba] merge
```bash
git status

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

```bash
git log
```

```
commit 80e0fba4ffa35b8f80e3f781a9282adccb89589e (HEAD -> master)
Merge: b9c0773 1333b20
Author: Drew Schmidt <wrathematics@gmail.com>
Date:   Tue Sep 20 13:14:01 2022 -0400

    merge
```

Wrapup

Wrapup

- Mistakes happen. It's not the end of the world.
- Staging/editing problems are probably the most common.
- Merge conflicts are inevitable when collaborating. Don't stress over it.
- Flight Rules for git https://github.com/k88hudson/git-flight-rules

Next Time

- git homework
- New module
- Basic Programming with R and Python
- Intro lecture

Questions?