Version Control

CSE 232 - Dr. Josh Nahum

Reading:

No Reading, but this is a nice tutorial: https://product.hubspot.com/blog/git-and-github-tutorial-for-beginners

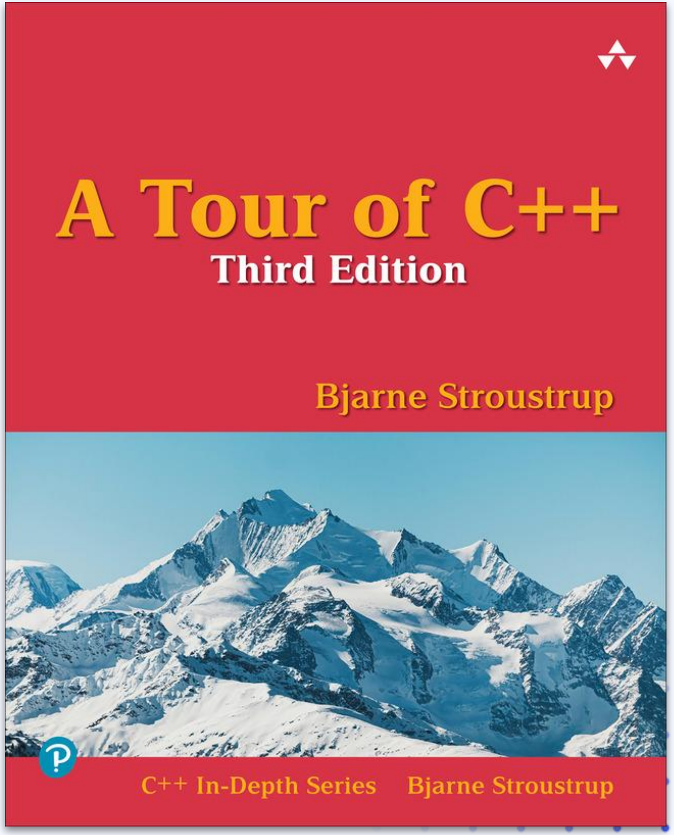


Table of contents

00

Motivation

01

git

02

Branches

03

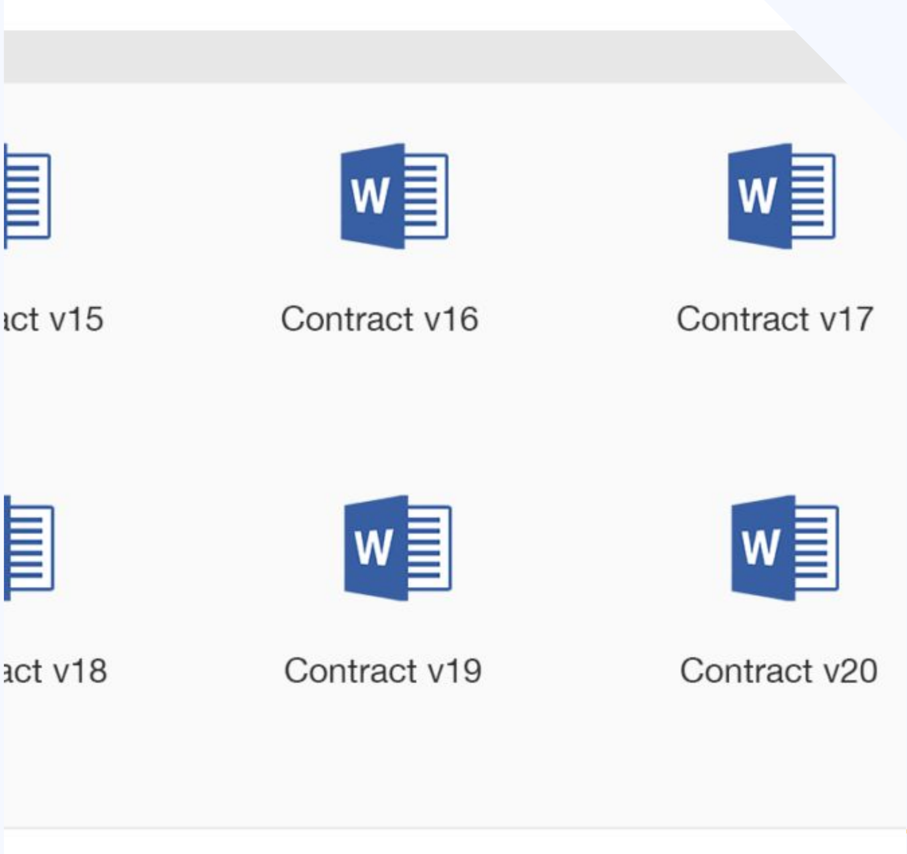
Collaboration

00

Motivation

Out of control versions

Have you ever many multiple versions of an important file so that you could undo changes if they break something?

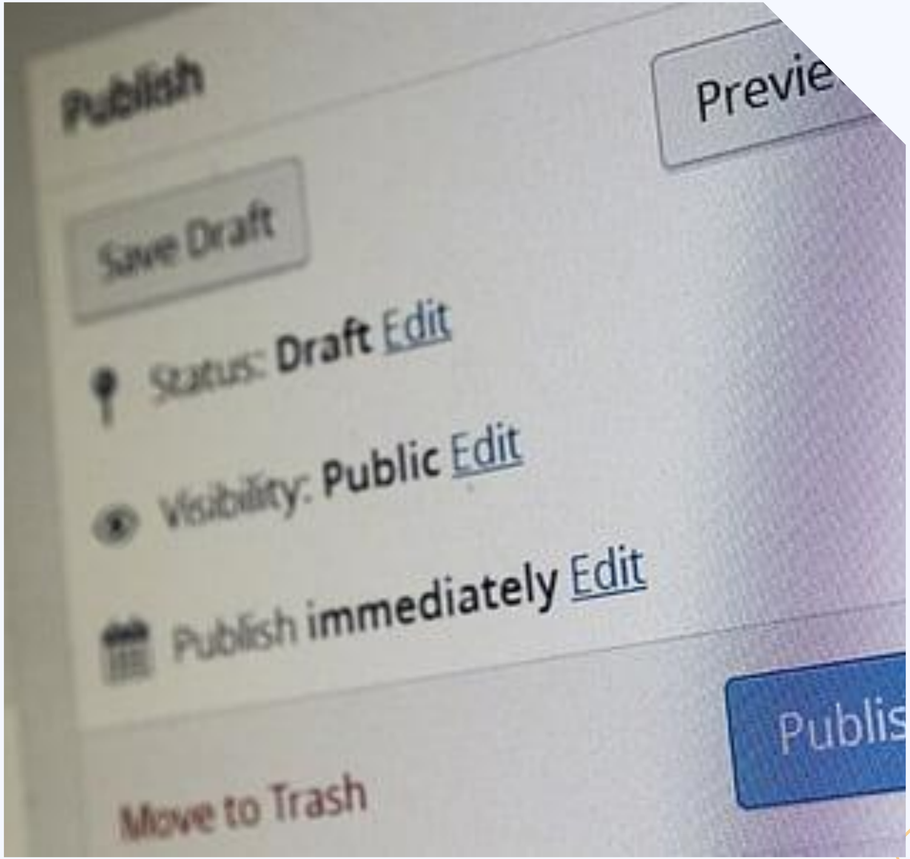


Multiple editors

Have you ever had a document that multiple people needed to edit simultaneously?

Publishing

Have you ever had a document that you needed to publicly share with others (possibly strangers)?



Status: Draft Edit

Visibility: Public Edit

Publish immediately Edit

“I can improve that process”

—git

01

git

Traits of git

Popular

git is the most common version control program

Open Source

git's source code is freely available (and managed by git)

Robust

git manages the most complex projects efficiently and correctly

Free

git is available everywhere and without cost

Terminology

Repository

A repo is a directory that is managed by git

Commit

The contents of a repo at a specific moment

Commit Message

A message about the changes made in a commit

Local Repo

This is the repo on your computer that you can change directly

Remote Repo

This is a repo hosted elsewhere that you can send commits to. Often hosted on sites like GitHub.

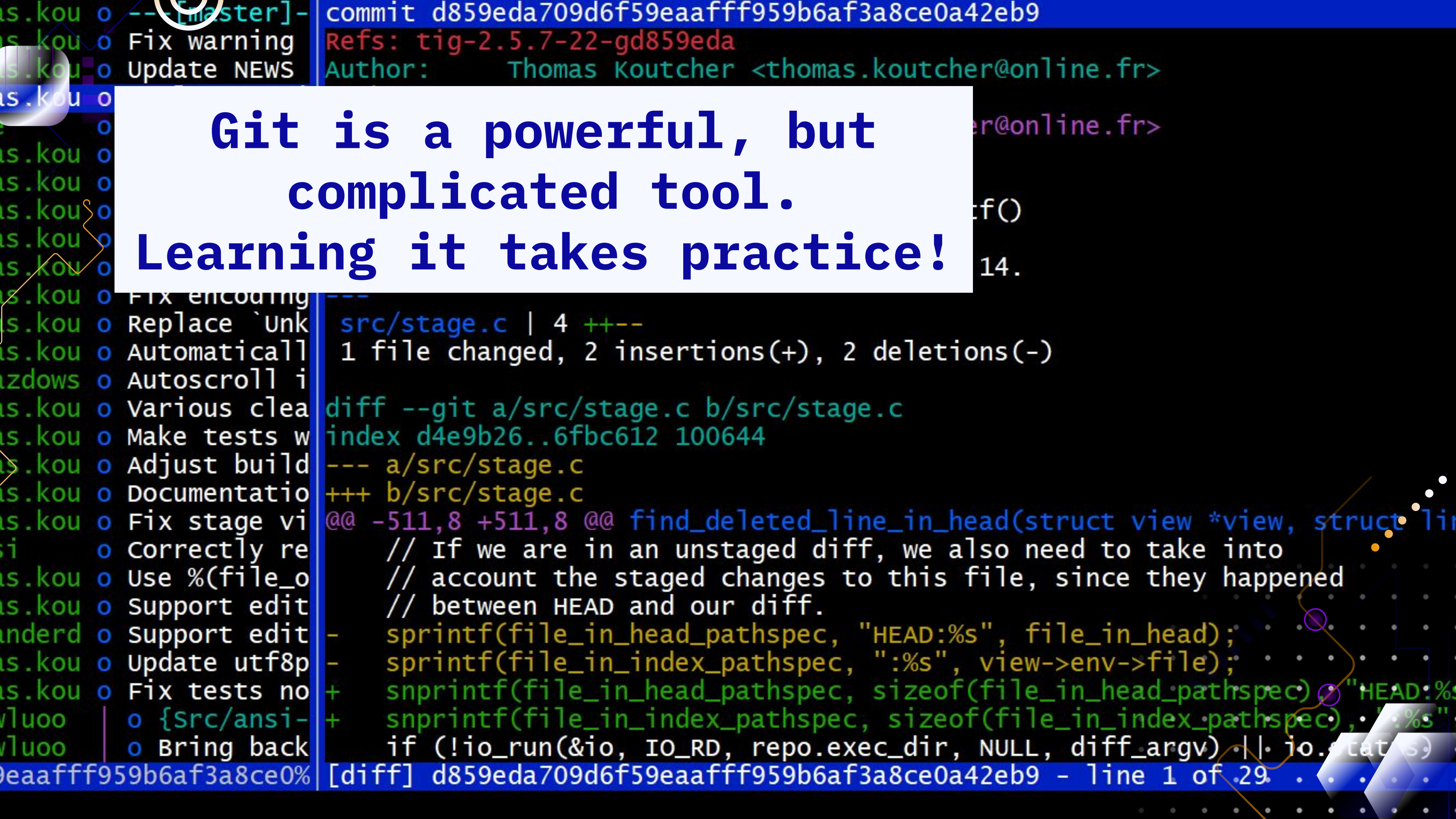
Branch

A named sequence of commits

Basic Commands

You should use a tutorial to get started, but here are some basic commands.

```
git init
# makes the working directory a git repo
git config --global user.name "A Student"
git config --global user.email student@msu.edu
# adds your identification to each git commit
git add .
# tells git to record all the changes to the repo in the current directory
git commit -m "Commit message"
# instructs git to record the changes in a commit with the given commit message
git log
# prints the list of commits
```



Git is a powerful, but complicated tool. Learning it takes practice!

```
s.kou o ---[master]-  commit d859eda709d6f59eaafff959b6af3a8ce0a42eb9
s.kou o Fix warning   Refs: tig-2.5.7-22-gd859eda
s.kou o Update NEWS   Author:     Thomas Koutcher <thomas.koutcher@online.fr>
s.kou o
      o                                                               er@online.fr>
s.kou o
s.kou o
s.kou o                                                               f()
s.kou o
s.kou o                                                               14.
s.kou o Fix encoding  ---
s.kou o Replace `Unk   src/stage.c | 4 ++--
s.kou o Automaticall   1 file changed, 2 insertions(+), 2 deletions(-)
zdows o Autoscroll i
s.kou o Various clea  diff --git a/src/stage.c b/src/stage.c
s.kou o Make tests w  index d4e9b26..6fbc612 100644
s.kou o Adjust build  --- a/src/stage.c
s.kou o Documentatio  +++ b/src/stage.c
s.kou o Fix stage vi  @@ -511,8 +511,8 @@ find_deleted_line_in_head(struct view *view, struct li
si    o Correctly re         // If we are in an unstaged diff, we also need to take into
s.kou o Use %(file_o         // account the staged changes to this file, since they happened
s.kou o Support edit         // between HEAD and our diff.
nderd o Support edit  -      sprintf(file_in_head_pathspec, "HEAD:%s", file_in_head);
s.kou o Update utf8p  -      sprintf(file_in_index_pathspec, ":%s", view->env->file);
s.kou o Fix tests no  +      snprintf(file_in_head_pathspec, sizeof(file_in_head_pathspec), "HEAD:%
luoo  | o {src/ansi-  +      snprintf(file_in_index_pathspec, sizeof(file_in_index_pathspec), ":%s"
luoo  | o Bring back         if (!io_run(&io, IO_RD, repo.exec_dir, NULL, diff_argv) || io.status)
eaafff959b6af3a8ce0%  [diff] d859eda709d6f59eaafff959b6af3a8ce0a42eb9 - line 1 of 29
```

02

Branches

Track Changes Improved

Branches

Independent commits can be made on different histories (branches) so that different work can be solved independently.

Checkout

Only one branch is active (checked out) at a time.

Branch Commands

```
git branch -l
# lists all of the branches and marks the checked out one

git branch my_branch
# creates a branch named "my_branch"

git checkout my_branch
# checks out the branch named "my_branch", commits will now occur
# on this branch

git merge other_branch
# combines the commits from branch named "other_branch" with this one

git branch -d my_branch
# deletes the branch named "my_branch"
```

Branches Aren't Always Necessary

Simple projects that don't involved collaboration don't have to use branches. But becoming proficient with them will make you a much more competent software developer.

For a fun way to practice and learn more about git branches, try this: https://learngitbranching.js.org/

03

Collaboration

Remote Repositories

Benefits

Having a repo that is visible to others allows other people to use your code and contribute suggested changes, without people having direct access to your computer.

GitHub

github.com is a common place for hosted repos. BitBucket, GitLab, and Launchpad are alternatives with similar functionality.

Remote Repos Commands

`git clone`

used to make a local remote from a remote repo

`git pull`

used to pull commits from a remote repo to the local one

`git push`

used to push commits from a local repo to a remote one

A Pull Request

A request from someone (usually on GitHub) to accept (pull) the commits they wrote

Attribution

Please ask questions via Piazza

Dr. Joshua Nahum
www.nahum.us
EB 3504

CREDITS: This presentation template was created by **Slidesgo**, and includes icons by **Flaticon**, and infographics & images by **Freepik**