Intro to Git

Joe Shields and Evan Yand

Who are we

PSAS: The Portland State Aerospace Society

- Evan
  - OreSat C3 Module
- Joe
  - Rocket airframe, and OreSat structure

What is Git?

- Created by Linus Torvalds in 2005 to aid Linux development
- **Distributed** Revision Control
- Github hosting for open-source projects
- Collaborate despite spotty communication/internet or conflicting strategies
- Other options: svn, cvs, BitKeeper, Visual Studio TFS

This Workshop

Will cover

- Basics of Git and Bash
- Using Git and GitHub to collaborate
- How to learn more

Will not cover

- How Git works
- Advanced features of Git
- Full use of Bash
- Use of the Git GUI

```
$ man giteveryday
```

covers some of these topics nicely.

Setup

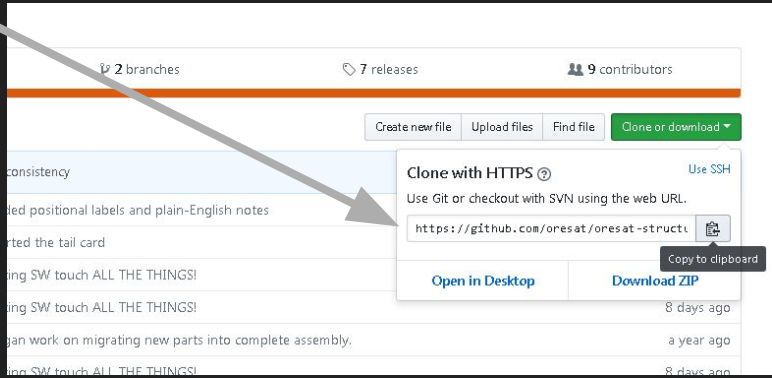
- Install
  - Debian-based: `sudo apt install git`
  - Windows: https://git-scm.com/download/win
- Check that it works
  - Open a terminal
  - `git --help`
- Create an account: https://github.com/join
- Get help
  - Linux: `man git` and `man git <command>`
  - https://git-scm.com/docs
  - https://www.kernel.org/pub/software/scm/git/docs
  - Search online, stack exchange, Youtube, et cetera...

Setup

- Set your username and email
  - `git config --global user.name "myGitHubUsername"`
  - `git config --global user.email johndoe@example.com`
- Set your editor
  - `git config --global core.editor nano`
  - Windows
- Check
  - `git config --list`

Create a Repo

- Using GitHub
  - Create a new repo from github.com (upper right corner.)
  - Follow the instructions.
  - `git clone <url to your repo>`
- Using only git
  - Create a directory with whatever name.
  - Go into that directory. (`cd nameOfYourRepo`)
  - `git init`



Using Git

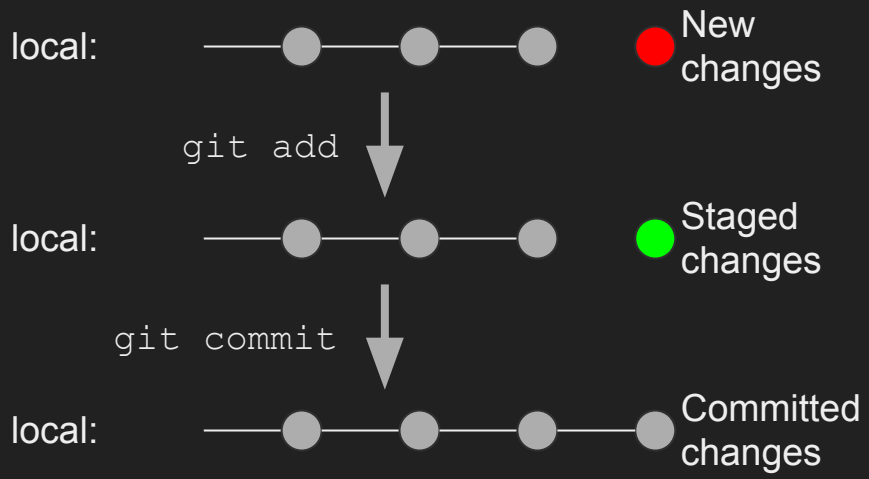
Basic Git Commands

- Clone
  - Copies a repository from a server to a directory.
- Pull
  - Fetches the remote copy of the repository from the server, and merges it with the local copy.
- Commit
  - Adds your changes to the local copy of the repository.
- Push
  - Updates the remote repository to match the local copy.

Clone

Copy the entire history existing at some remote location.

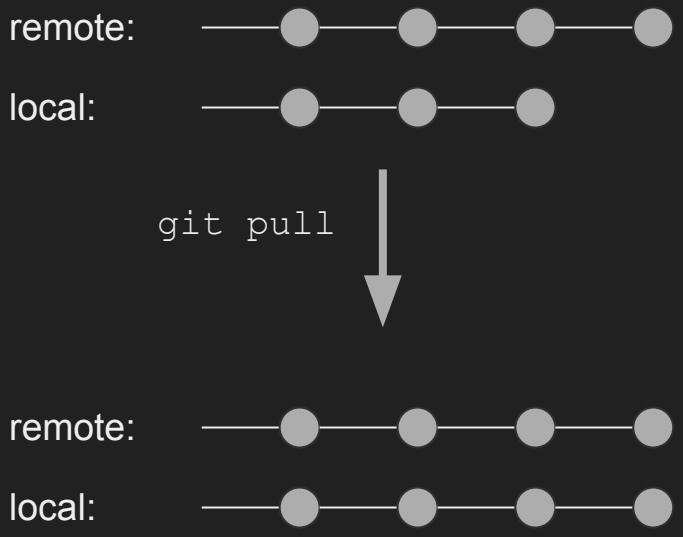
Add and Commit



Tell git what changes you want to `add` to a commit.

`commit` those changes to the history, so you can share them or see them later.

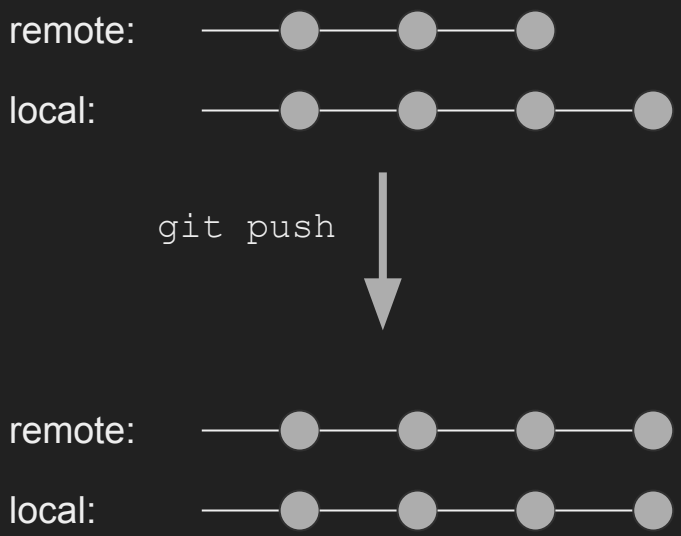
Pull



Get the new commits from [the master branch of] the remote.

(And, try to merge them with your history and then fast-forward to the most recent commit.)

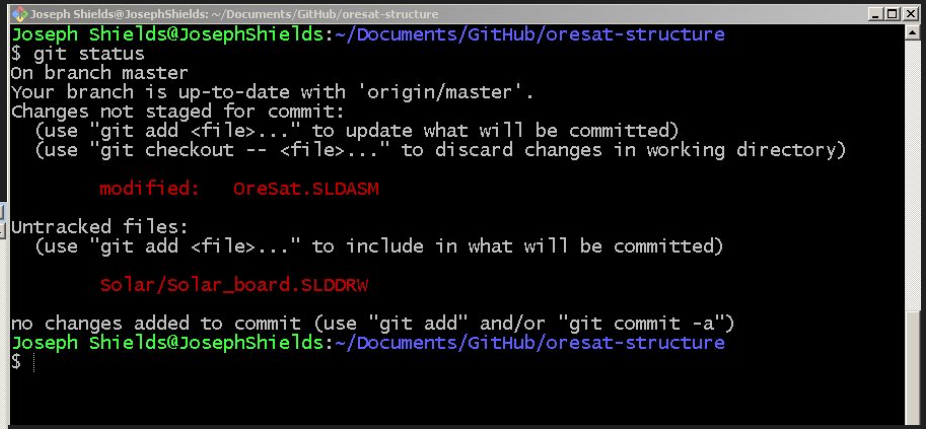
Push

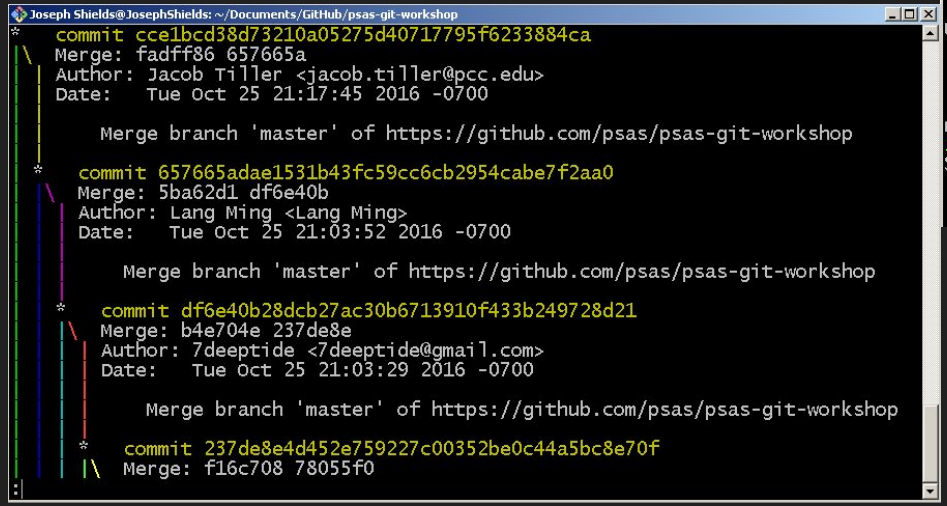


Update [the master branch of] the remote with your new commits [from your current branch].

Other Commands

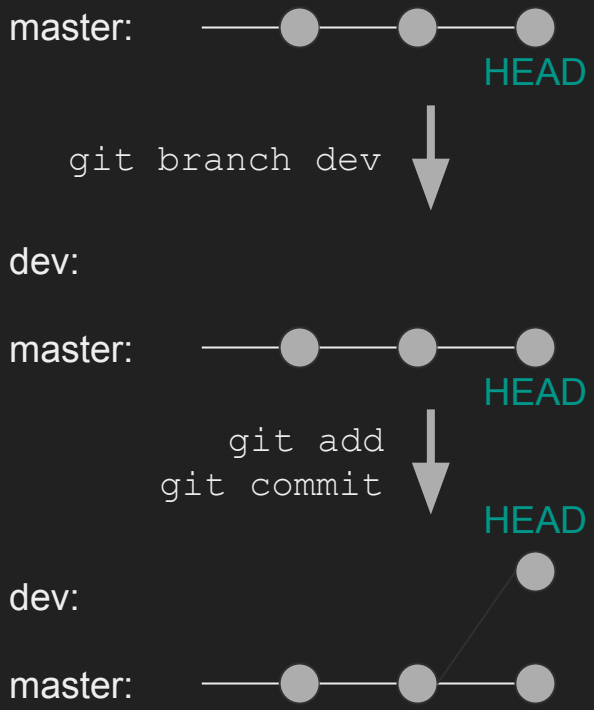
- `git status`
- `git log --graph`





Branches

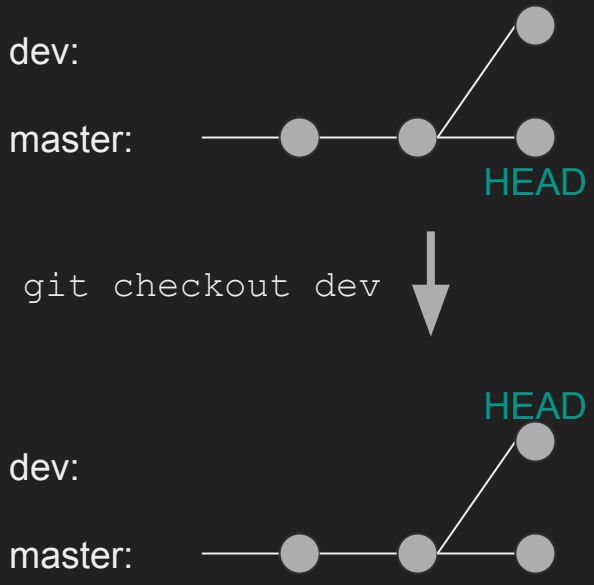
Create a branch



All your new commits will be on this new branch.

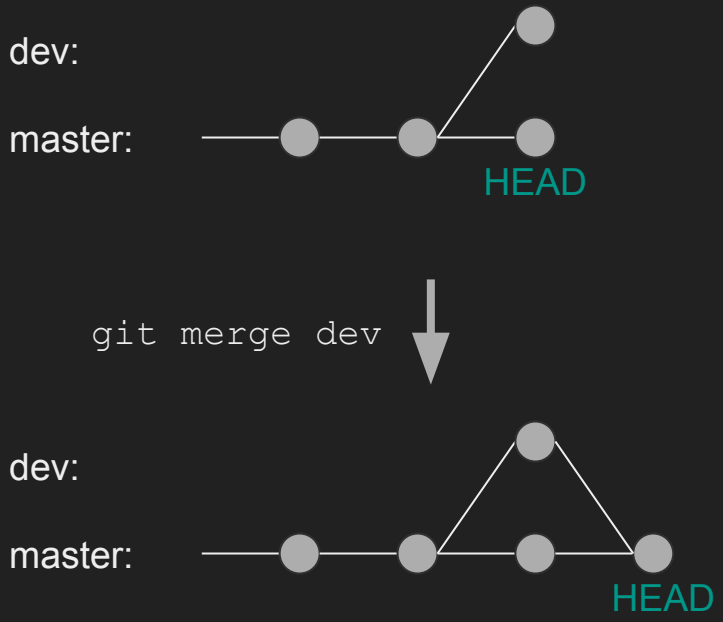
(You can see the available branches with `git branch -a`.)

Change Branches



Switch from one branch to another. Any new commits will continue from HEAD.

Merge Branches



Make a commit that incorporates the changes from another branch.

Merge Conflicts

How Conflicts Happen: Branches

- Line 123 of helloWorld.txt is changed on branch dev_evan.
- Line 123 of helloWorld.txt is changed on branch dev_joe.
- Joe tries to merge the changes from dev_evan, but Git doesn’t know whose changes should take priority.

How Conflicts Happen: Pushing

- Evan changes line 123 of helloWorld.txt on branch master and pushes to origin/master.
- Joe changes line 123 of helloWorld.txt on branch master and pushes to origin/master.
- Git tells Joe he needs to pull first, so he does.
- Git tries to merge the changes from origin/master with the local version of master, but Git doesn’t know whose changes should take priority.

Notice there’s a conflict

```
(trusty)joedang@localhost:~/Documents/hello_world
$ git merge dev_example
Auto-merging extra.txt
CONFLICT (content): Merge conflict in extra.txt
Automatic merge failed; fix conflicts and then commit the result.
(trusty)joedang@localhost:~/Documents/hello_world
$ git status
On branch mergeExample
You have unmerged paths.
  (fix conflicts and run "git commit")

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

        both modified:      extra.txt

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

Resolve the conflict

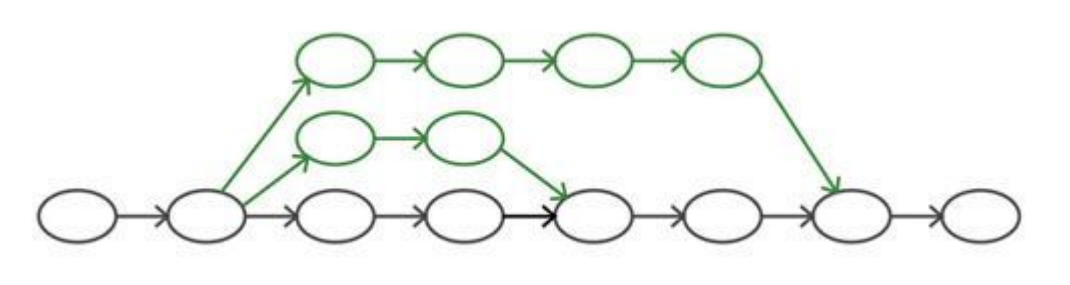
```
(trusty)joedang@localhost:~/Documents/hello_world
$ cat extra.txt
<<<<<<< HEAD
This is yet another file.
Here is a line I added to the file in the master branch.
=======
This is still another file.
I'm making changes to this line, but in a different way and on branch
dev_example.
Wheeee, more text!
>>>>>>> dev_example
(trusty)joedang@localhost:~/Documents/hello_world
$ vim extra.txt
(trusty)joedang@localhost:~/Documents/hello_world
$ cat extra.txt
This is yet another file.
Here is a line I added to the file in the master branch.
I'm making changes to this line, but in a different way and on branch
dev_example.
Wheeee, more text!
```

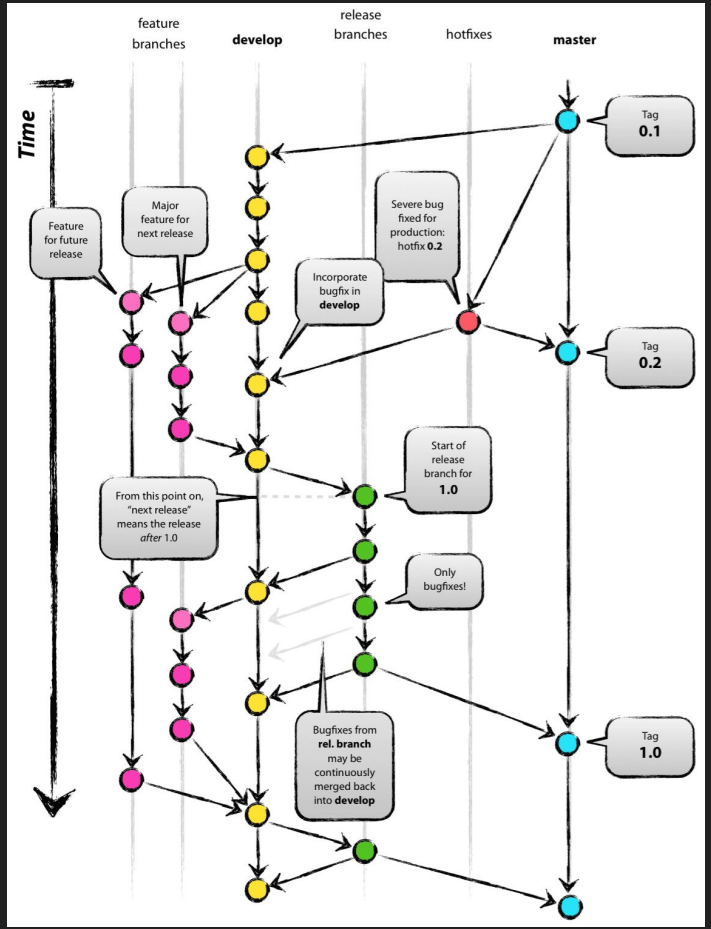
Commit your merge

```
(trusty)joedang@localhost:~/Documents/hello_world
$ git add extra.txt
(trusty)joedang@localhost:~/Documents/hello_world
$ git status
On branch mergeExample
All conflicts fixed but you are still merging.
  (use "git commit" to conclude merge)

Changes to be committed:

        modified:   extra.txt

(trusty)joedang@localhost:~/Documents/hello_world
$ git commit
[mergeExample a60b093] Merge branch 'dev_example' into mergeExample
(trusty)joedang@localhost:~/Documents/hello_world
$ git status
On branch mergeExample
nothing to commit, working directory clean
```

Advanced Things

Branching Methodology





GitHub features

- Bug tracking
- Wiki
- Repos/Organizations

Closing Tips

- Use man pages, cheat sheets, and the internet.
- Use Git on your own projects. Learn things as you need them.
- Only memorize:
  - `man`
  - `git status`
  - `ls`
  - `cd`