

A-Very-Hacktober Introduction to Open Source

Michael DeMarco

October 24th, 2023

A-Very-Hacktober Introduction to Open Source

- Welcome to this workshop on contributing to OSS!
- We'll get started shortly. . .

Introduction

- I'm Michael!
- ~~4th~~ 5th year Honours Computer Science and Data Science minor
- Former intern at the CSA, Tesla, Amazon; Academic Officer with the CSS
- @michaelfromyeg on the Internet

Agenda

- Introduction to open-source (5min.)
- Basic commands; a first contribution (25min.)
- Hacktoberfest (5min.)
- Break-out and exploration (15min.)
- Q&A, wrap-up (10min.)

Goals

- Get an understanding of open source (what it is, why it exists)
- Know enough `git` to confidently contribute to open source
- Learn about Hacktoberfest and look at making some significant contributions this month
- Meet other folks to work on open source projects with!

Expectations

- Some familiarity with writing programs beyond CPSC 110 (i.e., with something other than Racket)
- Some experience with `git` (though I can cover it, if need be!)

Getting started

- So... what the heck is this open source thing?!

Your turn: what do you *think* open source
is?

- Go ahead!

What is open source? (1/)

- Software that is **publicly accessible** so that the code can be seen, modified and use
 - This is restricted by a “license” (a bit like copyright)
 - More accurately, this is “free and open source” or FOSS; open source is focused on the visibility of the source code
 - “Free as in freedom, not as in beer”
- Projects often hosted on GitHub, GitLab, or similar in public repositories

What is open source? (2/)

- Many common examples: [Linux](#), Visual Studio Code, React, Flutter, Kubernetes, . . . countless others
- Companies even being built around open source (e.g., Mattermost vs. Slack, Supabase vs. Firebase)

Surprise Linus!



Figure 1: Linus Torvalds... founder of Linux!

Why contribute?

- Give back to software you use
- Improve your skills on a significant project (. . . though personal projects are cool too!)
- Find mentors
- Grow a reputation
- . . . get a job!

Maybe... even a job at Facebook

- Take a look at [this](#) article



Figure 2: Jumping in GIF

What does contributing look like?

- While it may be code, there are lots of other ways to give back
- Reporting issues, improving docs, suggesting features, reviewing PRs, etc., are all valuable

Joining a project (1/)

- Open source is based around the idea of community, and to be successful, you should know the language of the community!
- Community of **maintainers** and **contributors**
- Created by an **author**

Joining a project (2/)

- You should be aware of the LICENSE and README.md
- CONTRIBUTING.md is a document outlining how to get involved
- CODE_OF_CONDUCT.md outlines appropriate behavior
- Look for custom issue trackers (outside, say, GitHub), online communities (Slack, Discord)

Finding a project

- Your best bet... projects you already use!
 - PrairieLearn! We'll take a look at their issues today
- Look for small things, too
 - ~28% of casual contributions are documentation (like typo fixes)
- Variety of resources online: [Good First Issue](#), [GitHub Explore](#), many more

Before you contribute. . .

- Check the following things
 - What's the license?
 - Does it accept contributions?
 - Is it active?
 - Are there open issues? How fast are folks getting responses?
 - Same for pull requests!
 - Are the maintainers friendly?

Our first contribution

- Let's make a quick detour... it's git time



Figure 3: Detour

Why use git?

- Open-source depends on community, so we need a sane way of managing versions of software
- Enter: version control!
- git is a command-line tool for managing versions of software projects, efficiently
- (Also authored by Linus Torvalds!)

Git for contributing to open source

- (Note... this is not a git workshop; we'll just do enough to make you functional)
- First, make sure `git` is installed
- (If on Windows, please use Chocolatey)

Configure git

- Run commands to inform who the contribution will be from

```
git config --global user.name "Your Name"
```

```
git config --global user.email "youremail@domain.com"
```

Fork the repository

- To make changes **to a repository we can't edit directly**, we make a fork
- This is do-able right within GitHub

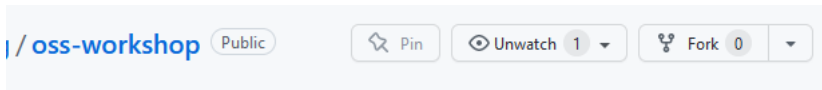


Figure 4: Fork repository on GitHub

After the fork

- Now you have a copy of the project to work in



Figure 5: After the fork

Clone the project (1/)

- Next step... let's get that fork on your machine!
- This will create a directory on your machine under `oss-workshop` (or whatever the repository name is)

Clone the project (2/)

with HTTPS

```
git clone https://github.com/getrooter/oss-workshop.git
```

or, with SSH

```
git clone git@github.com:getrooter/oss-workshop.git
```

or, with the GitHub CLI

```
gh repo clone your-username/oss-workshop
```

Open the project in your editor

- You'll now need to make some changes!
- First, open the project in an editor
- (I recommend Visual Studio Code, a lot)

```
cd ~/oss-workshop  
code .
```

Make a branch (1/)

- Features or bugs that are non-trivial often take a lot of time, and other developers will continue making changes meanwhile
- To create an isolated space for our changes, we'll create a feature branch

Make a branch (2/)

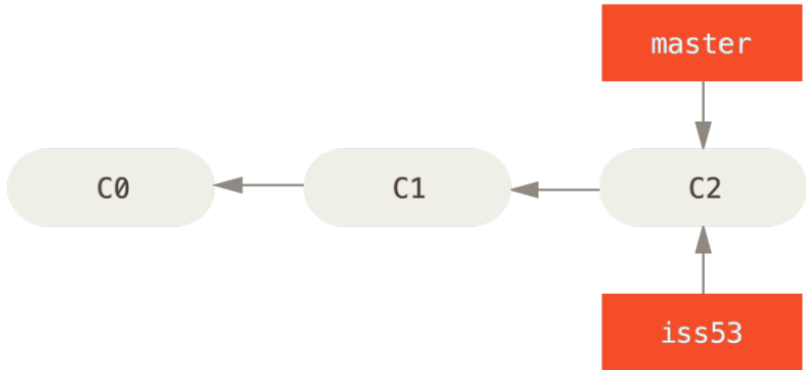


Figure 6: Creating a branch called `iss53`

Make a branch (3/)

- The commands to do this are below; make sure to check community conventions for the branch name!

verify you're on main

`git status`

make the branch

`git checkout -b my-descriptive-branch-name`

Make your changes. . .

- Your turn! Add yourself to the README.md of this very repository
- Bonus points: edit src, add open source repositories to docs/IDEAS.md
- You'll have about 10 minutes to do this

Commit your changes

- Now that your changes are ready, let's commit them to your branch

verify which files changed

```
git status
```

```
git add .
```

```
git commit -m "a descriptive message"
```


A brief note on commit messages

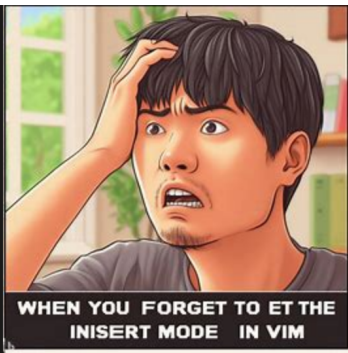
- Sometimes, you'll want a longer commit message
- Try setting an editor, and you can have a multiline commit by running `git commit` (no `-m`)

```
git config --global core.editor "nano"
```

```
# or
```

```
git config --global core.editor "vim"
```

vim



Push your changes

- Our changes exist on our local branch; we need them to exist in a remote branch
- This is called “pushing” changes (so a remote origin, some other git server)

```
git push -u origin my-descriptive-branch-name  
# -u is short for --set-upstream
```

Open a pull request (1/)

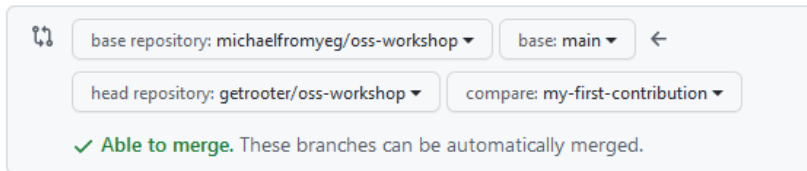
- We're now moving from `git` to GitHub; we want to use the browser to open a pull request (PR) against the original repository
- The goal was to merge our change in after all

Open a pull request (2/)

- Hit the new request button on GitHub
- Make sure to include the required information (again... community conventions!)

Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to,



The screenshot shows the GitHub 'Comparing changes' interface. It features a light blue header with a fork icon on the left. Below the header are four dropdown menus arranged in two rows. The first row contains 'base repository: michaelfromyeg/oss-workshop' and 'base: main', followed by a left-pointing arrow. The second row contains 'head repository: getrooter/oss-workshop' and 'compare: my-first-contribution'. Below these dropdowns, a green checkmark icon is followed by the text 'Able to merge. These branches can be automatically merged.'

base repository: michaelfromyeg/oss-workshop ▼ base: main ▼ ←

head repository: getrooter/oss-workshop ▼ compare: my-first-contribution ▼

✓ **Able to merge.** These branches can be automatically merged.

Figure 8: Compare changes across branches

Open a pull request (3/)

- Take a look at [my example PR](#)!
- ...we've done it!

Beyond the PR... (1/)

- You may have to make additional commits to get your PR prepared based on feedback; that's okay!
- There's some additional `git` concepts you'll want to be familiar with...
- ...but they're beyond the scope of this workshop

Beyond the PR... (2/)

- Your homework...
 - `git commit --amend` ([here](#))
 - Resolving merge conflicts ([here](#))
 - Merge vs. rebase ([here](#))
 - Squashing commits ([here](#))
- If you have questions: the Internet, your friends, or me!
- CSSS git workshop is coming up soon!

Your first contribution

- Let's get this attendance list filled out! Now's the time... make your PRs!
- (Also try opening an issue!)

Hacktoberfest (1/)

- Hacktoberfest is a chance to support open source with the world!
- ... and ~~get a free t-shirt~~ plant a tree!
- Head to hacktoberfest.com and sign-up with GitHub

Hacktoberfest (2/)

- You ~~must make 4 PRs to get a t-shirt~~ only need 1 PR to plant a tree!
- **Warning:** do not make low quality PRs, don't force the ~~shirt~~ tree!
- Consider co-working with your friends; hold each-other accountable

Hacktoberfest (3/)

- There's also events! ...including this!
- Group photo!
- Get engaged at @hacktoberfest, @digitalocean, #hacktoberfest, #hacktoberfest2023

Hacktoberfest (4/)

- If you do hit all 4, please let me know! I'd love to recognize how many folks are able to get their within our lil' UBC community



Suggested repositories

- [PrairieLearn](#)
- [csss-site](#)
- [ubyssey.ca](#)
- Other ideas?

Breakout

- Your turn: go explore these (or other) repositories, and see if any issues jump out at you
- (Optional) I can walk through setting up PrairieLearn locally, since it's a bit tricky to get it all going
 - (We may not finish. . .)

Network

- Let's find other folks to work on open source with
- Say hi to your neighbors and see what project(s) their interested in
 - (And make sure to stay connected via social media!)

Q&A

- Questions?!

Wrap-up

- Today, we
 - Learn what open source is
 - Explored the basic commands to create a PR
 - Got onboarded to Hacktoberfest
 - Discovered a few candidate repositories
 - Met some like-minded folks!
- Beyond the workshop
 - Learn some more `git`
 - Keep contributing!

Thanks for coming!

- Stay in touch @michaelfromyeg on the Internet or mdemar01 [at] student [dot] ubc [dot] ca