Open Source 101

Justin W. Flory

What are we covering today?

Agenda

1. What is open source?
    a. Quick gist of what it is and what it means
2. Open source beyond software (ft. Dan Schneiderman)
3. Open source your education (ft. Prof. Stephen Jacobs)
    a. Free and Open Source Software Minor at RIT
4. Getting git
5. Contributing to open source projects
    a. Finding a project and finding a good one

So, uh… what is open source?

At a literal glance

- Often referred to as **FOSS**: free and open source software
  - **Free as in speech**, not free as in pizza
  - Source code **publicly available** for reading, use, modification
- Term is used to not only to describe software, but the **development practices, licensing, and communities** that help build the software
  - Common set of values
- **Where is open source?**
  - Your pocket
  - This room
  - Powering the Internet

Beyond the code

- **Open source:** Public code, but *why*?
  - **Control**: Examine the code, “have it your way!”
  - **Training**: Study guides! Get feedback. Share mistakes.
  - **Security**: More eyes on code, can be quicker to fix things
  - **Stability**: Open standards, if the maintainer leaves, someone else can lead
- Timeline
  - **1983**: Richard Stallman, GNU project
  - **1985**: Free Software Foundation (FSF)
  - **1989**: GNU Public License (GPL)
  - **1997**: *The Cathedral and the Bazaar*
  - **1998**: Open Source Initiative

Four Freedoms of Open Source

These four principles (or four R's) are part of what makes "free and open source software" free and open source

- **Read:** Freedom to read the code
- **Run:** Freedom to run the code any way you like
- **Revise:** Freedom to make changes to suit your needs
- **Redistribute:** Freedom to redistribute your changes to others

Linus's Law: *Given enough eyeballs, all bugs are shallow*

Examples of open source

- **On the web:**
  - Linux: Powering ~79.3% of the web's infrastructure
  - Apache HTTPD Server: "The Number One HTTP Server On The Internet"
  - Web technologies like Node.js, Flask, Ruby on Rails
- **In your pocket:** Android, Reddit… teleirc!
- **On your laptop:** Firefox, Chromium, SpigotMC (Minecraft)
- **In your code:** Python, Swift, Dotnet

“…open source is an **intellectual property destroyer**. I can't imagine something that could be **worse than this** for the software business and the intellectual-property business.”

- Microsoft, 2001

“Microsoft has been working with open source for a while - **over ten years already**. It started with support for Novell and PHP. [...] Last year, Microsoft CEO Satya Nadella publicly declared the **company's love for Linux**, and he's remained true to his word since. Microsoft is now involved with **140 workgroups dealing with open standards**, and actively supports more than **400 projects** where code is written before being given back to the community.”

- Microsoft, 2016

Dan Schneiderman

FOSS@MAGIC Research Associate & Community Liaison at RIT

Website

Open source goes beyond just code…

Prof. Stephen Jacobs

Professor; RIT MAGIC Center Research Affiliate; Visiting Scholar, National Center for the History of Electronic Games

Open source your education!

Introducing the Free and Open Source Software Minor at RIT

Getting `git`

Git? What? What are we getting?

- **git**: free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency
- English?
  - Save points in a video game
  - With git, you can revert back to a previous time in your code
  - Also simplifies working collaboratively with others asynchronously
  - Imagine a ledger of all revisions to your code, like a list of save points

Application: Homework assignment

- Imagine you're working on a homework assignment
  - Each time you add a major feature or hit a part of the rubric, you "commit" your code with git
- You decide to be ambitious and go for the bonus points
  - Uh-oh, you accidentally the entire assignment
  - It's 1am and you want to go back to where you were three hours ago because that met the rubric requirements and was technically done
- Revert back to a specific commit with Git
  - Like magic, your code is restored to that point in time
- Hooray! You can go to sleep with a completed homework assignment and secret disappointment over not getting the bonus


Truegif.com
git push origin master --force

“Tag” milestones of your code

- Another handy feature of Git is tagging
  - You can “tag” a commit with as a specific point in history, like a version number
- Create a “tag” as an easily referenceable milestone in your code
  - For example, a personal project you are working on
  - You get it working and minimal functional, hooray! v1.0
  - Later, you hope to expand the features and add some other things
  - Skip ahead in the future, yay! It’s ready. v1.1
  - Both tags are easily reference for later

Working collaboratively

Git is great at code parties

- Git makes collaboration with others easier
  - Git can handle commits from multiple “authors”, automatically merge them if there are conflicts (or make it easier to merge them manually if it can’t automatically)
  - More convenient than copy+pasting code among your team members
- Introducing pull requests *
  - Great way to handle **code review**
  - Partner 1 works on project and has code ready
    - Opens pull request against Git repo (i.e. your code homebase)
    - Pull request has the code viewable by anyone but it is not actually “in” the main code
    - Once teammates review and look over it, they can accept and merge, or reject the pull request

What did the * mean?

- A note about pull requests
  - Not really an “actual” feature of Git
  - More of a front-end tool

Where does GitHub fit in?

- First, know this: Git is just software
  - Anyone can download and install git on their computer
  - It’s the actual powerhouse behind this whole thing
- GitHub is a public place for you to host your git repositories online
  - GitHub also adds a fancy web presence for your project on their website
  - Easy to browse code online and sometimes even make small edits
  - Also comes with unique tools like issue trackers, wikis, and project website hosting (for free!)
- GitHub is only a **front-end** for git

In the real world

Why bother with this git thing?

- Outside of using Git to help make your projects and homeworks easier, it is widely in use
- Git is an **industry standard** for version control
  - Likely to experience it eventually while you're at RIT
  - After graduating, if you write code in an organization or company, you are almost guaranteed to find this
  - It's literally everywhere
- Learning this now and applying it to "small" things like homeworks or projects is making your future self's life easier

“I can just learn this later.”

- You can learn it later, but employers are already looking to see if you know it
- Not uncommon for employers / interviewers to ask for your **GitHub profile**
  - More about GitHub in a moment
- Learning Git and open sourcing your code is a great way to show off your experience and knowledge for that co-op you really want
  - Also has an ethical aspect about **open source** and what exactly that means for you and your code

Git is going to get
you sooner or later.

Confused? Lost? Wondering how to actually do this?

- Worry not!
  - Git is everywhere, thus…
  - **Countless websites, guides, and documentation** teaching git
- By default, Git is a command line utility
  - If you're just getting started, can be intimidating
  - The GitHub Desktop app is a great place to get started (desktop.github.com)
    - Has an easy-to-use, understandable, and functional GUI for interacting with Git and GitHub

Contributing to open source

Before going further…

- Open source is something **absolutely realistic for students to pursue!**
  - Can be tough and confusing at first
  - But it's not as hard as it seems from the outside!
- There are many different projects of varying sizes - **difference processes** for getting involved
- Here's some tips for **getting started with FOSS** and what opportunities come with it!
  - **Two big criteria** for the search

(1) Finding a project that interests you

- MOST important part of contributing to open source is contributing to something that **interests *you***
  - Interest drives motivation and drives your likelihood to succeed
- **Wrong way** to approach it is choosing a project based on **connections you *might* make** or trying to **get your name out**
  - Important, but will come in time
- Look at your computer, your phone: **what's something you like?** Is it open source? Are they using open source software?

(2) Aligning personal experience with interests

- After finding a project, compare the project **your own experience and knowledge**
  - You **don't have to be an expert** - but enough to break in somewhere is enough
- Most obvious thing is **programming language**: Python, Java, JavaScript, Ruby, Rust, GoLang… the list goes on
  - There are **more "hats"** than just programmers
- Open source **needs** designers, community people, marketing people, and writers as much as programmers!

Before jumping in, evaluate

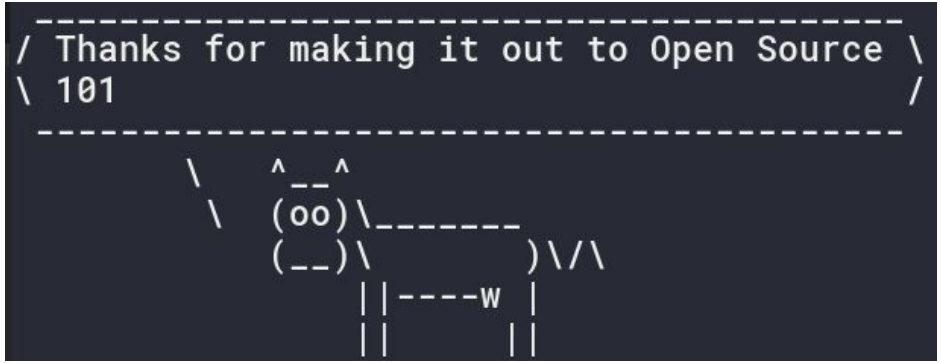
- Evaluating the project before immersing yourself is important: **consider the community**
  - Important to choose a community that will support you contributing to their project and acknowledge that you are a student
- Read mailing lists, chat logs, hang out in development rooms… **get an idea** of what the community is like
  - **Inclusivity** and **positive engagements** are characteristics to look for
- Also: look for helpful tips to get starting (a.k.a. **on-boarding materials**)

Benefits of contributing

- **Apply what you're learning to real-world projects**
  - Gain experience and build your skills further - and have something to point to
- **Networking**
  - Meet brilliant people from across the tech world
- **Leadership opportunities**
  - Chance to help lead on something you are passionate about
- **Your perspective matters**
  - More students and young people need to be in open source
- **Friendships and travel opportunities**

Questions? Comments? Concerns?

Author: Justin W. Flory
Date: Fri Dec. 2 2016



Don't forget to sign in if you didn't already:

signin.ritlug.com

Stickers available on request!

References

- https://opensource.com/resources/what-open-source
- https://opensource.com/life/15/12/why-open-source
- https://en.wikipedia.org/wiki/Free_and_open-source_software
- https://www.gnu.org/licenses/gpl.html
- https://opensource.com/life/16/10/my-open-source-story-justin-flory