

# Open Source

Tapasweni Pathak





# Something about me..

- Contributed to Linux Kernel as Outreachy Intern.
- Contributed to OWASP Hackademic as its Summer Code Sprint Student.
- Minor patches in Debian Sources, X.org.
- Mentor GSOC'15 Syssters Django based projects.
- Mentor GCI'15 Syssters Django based projects.
- Lead Project Mentor GSOC'16 Syssters Django based projects.
- Maintainer Syssters Volunteer Management System Project.
- Have open sourced a lot of my side projects.
- Writes (a lot) and speaks (sometimes) about getting involved with open sources.



# Open Source?

- Open-source software (OSS) is computer software with its source code made available with a license in which the copyright holder provides the rights to study, change, and distribute the software to anyone and for any purpose. Open-source software may be developed in a collaborative public manner. Open-source software is the most prominent example of open-source development.
- The open-source model, or collaborative development from multiple independent sources, generates an increasingly more diverse scope of design perspective than any one company is capable of developing and sustaining long term.

--Wiki



# Licenses

- Default rule says you own your code completely this means that you retain all rights to your source code and that nobody else may reproduce, distribute, or create derivative works from your work.
- Sharing your code also includes telling people how they can use your code.



# Types of Licenses?

- **MIT License : Simple and Permissive**

You can do anything you want with my code as long as you provide attribution back to me and don't hold me liable.

- **Apache License 2.0 : Concerned about Patents**

Distribute, modify. Original copyright, patent, trademark and attribution notices must be preserved.

- **GNU General Public License v3 : Care about sharing improvements**

Derived works can only be distributed under the same license terms.

# ChooseALicense.com

## Choose an open source license

{ Which of the following best describes your situation? }



### I want it simple and permissive.

The [MIT License](#) is a permissive license that is short and to the point. It lets people do anything they want with your code as long as they provide attribution back to you and don't hold you liable.

[jQuery](#), [.NET Core](#), and [Rails](#) use the MIT License.



### I'm concerned about patents.

The [Apache License 2.0](#) is a permissive license similar to the MIT License, but also provides an express grant of patent rights from contributors to users.

[Android](#), [Apache](#), and [Swift](#) use the Apache License 2.0.



### I care about sharing improvements.

The [GNU GPLv3](#) is a copyleft license that requires anyone who distributes your code or a derivative work to make the source available under the same terms, and also provides an express grant of patent rights from contributors to users.

[Bash](#), [GIMP](#), and [Privacy Badger](#) use the GNU GPLv3.

{ What if none of these work for me? }

### My project isn't software.

[There are licenses for that.](#)

### I want more choices.

[More licenses are available.](#)

### I don't want to choose a license.

[You don't have to.](#)

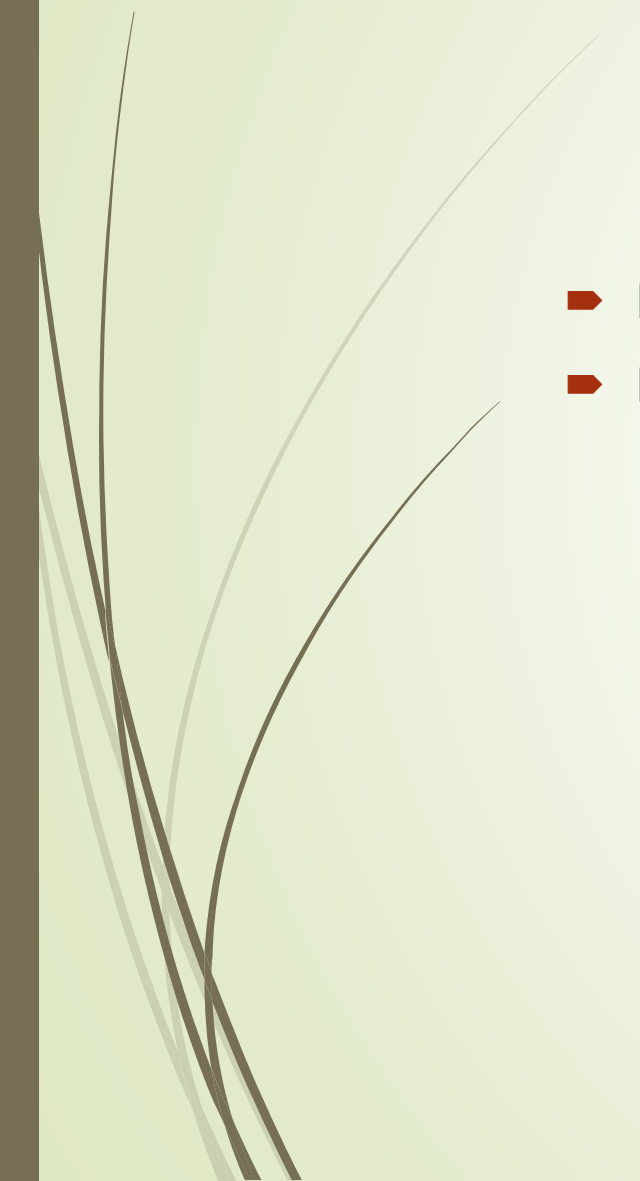


# Open Source Software and Proprietary Software





# Why Contribute?

- Help yourself.
  - Help others.
- 





# How to?

- Mailing List and IRC.
- Documentation.
- Issue Trackers.
- Source Code.
- Setting up the development environment.
- Coding practices.
- Testing Code.
- Documenting your code.
- Semantic Versioning.
- Requesting merge and discussion.



# Mailing List and IRC



# Documentation



# Setting Up the Development Environment



# Source Code



# Coding Practices



# Testing





# Documenting Your Code



# Issue Trackers



# Semantic Versioning

- Given a version number MAJOR.MINOR.PATCH, increment the:
- MAJOR version when you make incompatible API changes,
- MINOR version when you add functionality in a backwards-compatible manner, and
- PATCH version when you make backwards-compatible bug fixes.
- Additional labels for pre-release and build metadata are available as extensions to the MAJOR.MINOR.PATCH format.



# Merges and Discussions



# Linux



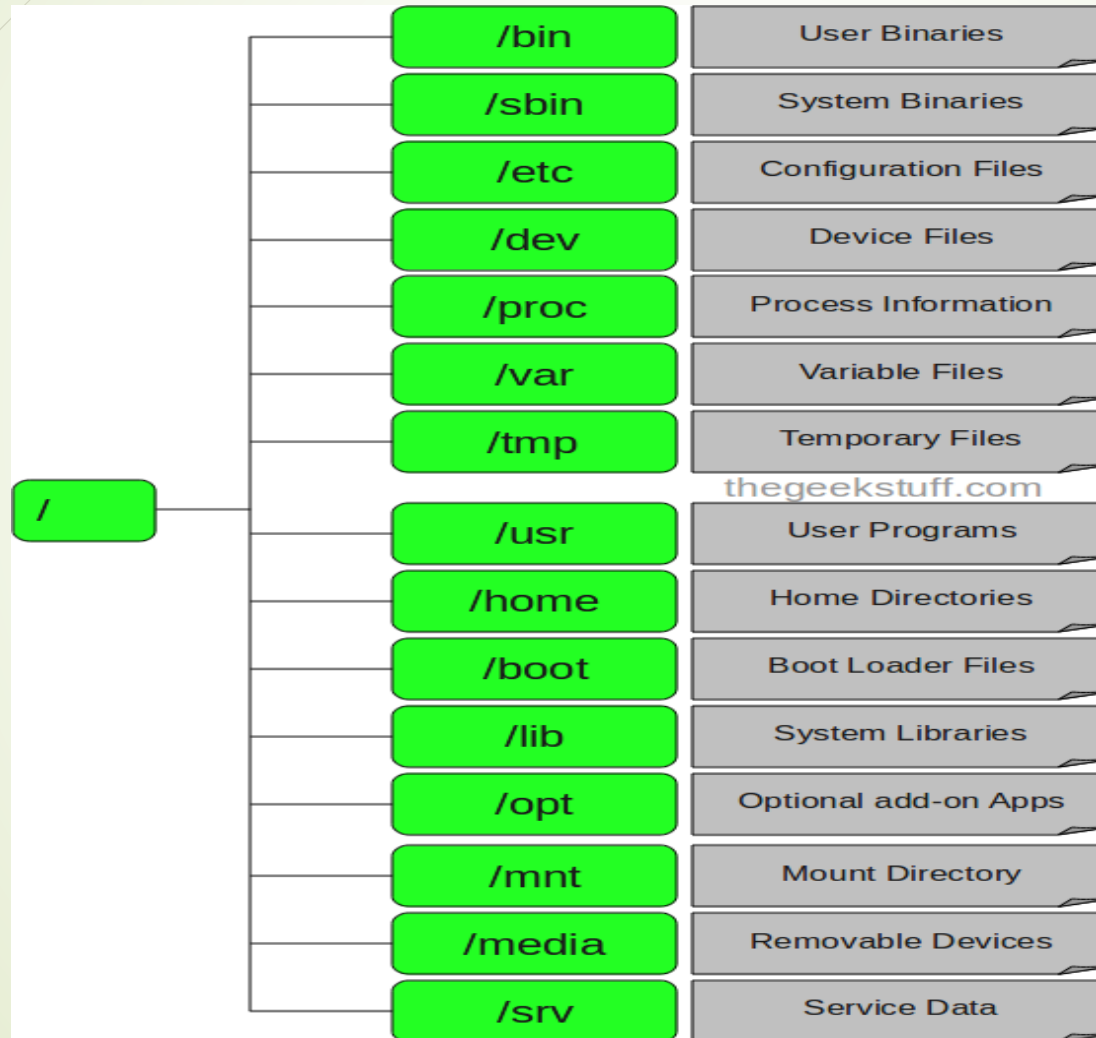
- A Unix-like and mostly POSIX-compliant computer operating system (OS) assembled under the model of free and open-source software development and distribution.

-- wiki



# Root, users and groups, and permissions

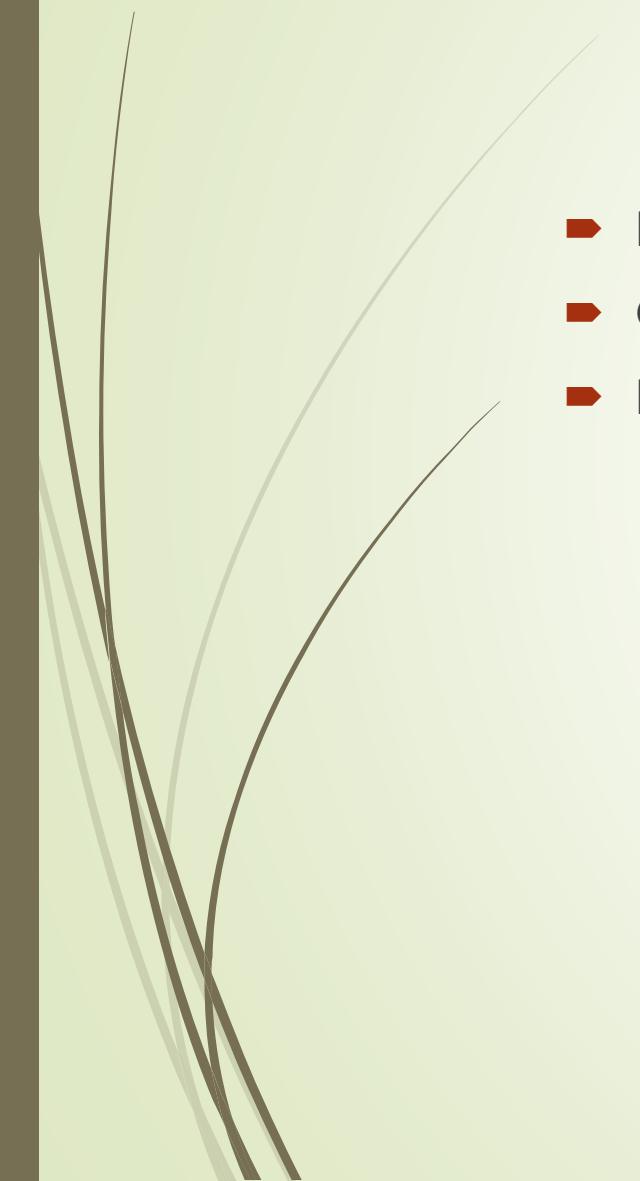
# File System Structure







# Root

- Every single file and directory starts from the root directory.
  - Only root user has write privilege under this directory.
  - Please note that `/root` is root user's home directory, which is not same as `/`.
- 



# /bin – User Binaries

- Contains binary executables.
- Common linux commands you need to use in single-user modes are located under this directory.
- Commands used by all the users of the system are located here.
- For example: ps, ls, ping, grep, cp.



# /sbin – System Binaries

- Just like /bin, /sbin also contains binary executables.
- But, the linux commands located under this directory are used typically by system administrator, for system maintenance purpose.
- For example: iptables, reboot, fdisk, ifconfig, swapon



# /etc – Configuration Files

- Contains configuration files required by all programs.
- This also contains startup and shutdown shell scripts used to start/stop individual programs.
- For example: `/etc/resolv.conf`, `/etc/logrotate.conf`



# /dev – Device Files

- Contains device files.
- These include terminal devices, usb, or any device attached to the system.
- For example: `/dev/tty1`, `/dev/usbmon0`



# /proc – Process Information

- Contains information about system process.
- This is a pseudo filesystem contains information about running process. For example: /proc/{pid} directory contains information about the process with that particular pid.
- This is a virtual filesystem with text information about system resources. For example: /proc/uptime



# /var – Variable Files

- ▶ var stands for variable files.
- ▶ Content of the files that are expected to grow can be found under this directory.
- ▶ This includes — system log files (/var/log); packages and database files (/var/lib); emails (/var/mail); print queues (/var/spool); lock files (/var/lock); temp files needed across reboots (/var/tmp);





# /tmp – Temporary Files

- Directory that contains temporary files created by system and users.
- Files under this directory are deleted when system is rebooted.



# /usr – User Programs

- Contains binaries, libraries, documentation, and source-code for second level programs.
- /usr/bin contains binary files for user programs. If you can't find a user binary under /bin, look under /usr/bin. For example: at, awk, cc, less, scp
- /usr/sbin contains binary files for system administrators. If you can't find a system binary under /sbin, look under /usr/sbin. For example: atd, cron, sshd, useradd, userdel
- /usr/lib contains libraries for /usr/bin and /usr/sbin
- /usr/local contains users programs that you install from source. For example, when you install apache from source, it goes under /usr/local/apache2



# /home – Home Directories

- ▶ ome directories for all users to store their personal files.
- ▶ For example: /home/john, /home/tapasweni



# /boot – Boot Loader Files

- Contains boot loader related files.
- Kernel initrd, vmlinuz, grub files are located under /boot
- For example: initrd.img-2.6.32-24-generic, vmlinuz-2.6.32-24-generic



# /lib – System Libraries

- Contains library files that supports the binaries located under /bin and /sbin
- Library filenames are either ld\* or lib\*.so.\*
- For example: ld-2.11.1.so, libncurses.so.5.7



# /opt – Optional add-on Applications

- opt stands for optional.
- Contains add-on applications from individual vendors.
- add-on applications should be installed under either /opt/ or /opt/ sub-directory.



# /mnt – Mount Directory

- Temporary mount directory where sysadmins can mount filesystems.





# /media – Removable Media Devices

- ▶ Temporary mount directory for removable devices.
- ▶ For examples, /media/cdrom for CD-ROM; /media/floppy for floppy drives; /media/cdrecorder for CD writer



# /srv – Service Data

- ▀ srv stands for service.
- ▀ Contains server specific services related data.
- ▀ For example, /srv/cvs contains CVS related data.



# Terminal Usage

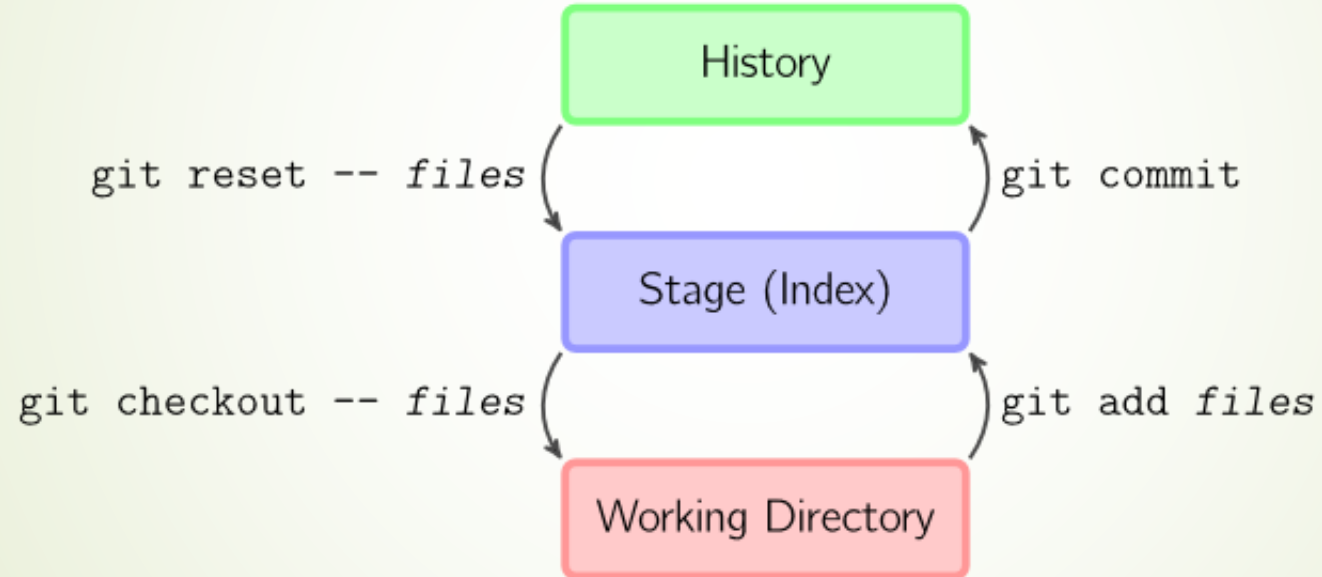


# Version Control



Git

# Basic Git





# GitHub



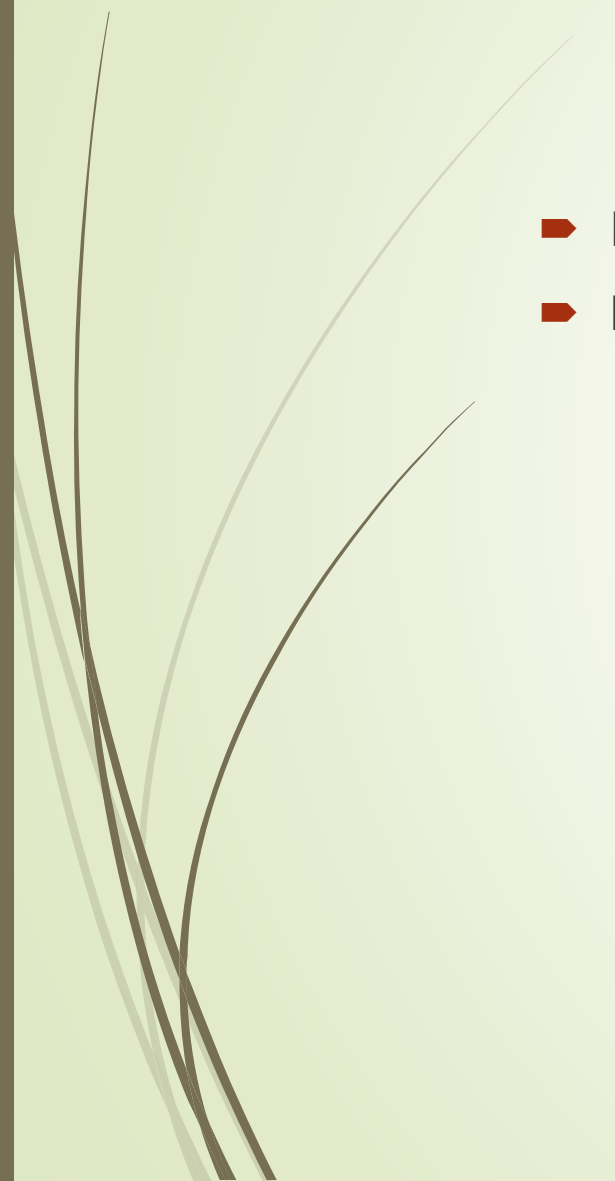
# GitHub Based Terminologies

- Repository
  - Pull Request
  - Fork
  - Issue
  - Star
  - Watch
- 





# Some Terminologies

- Patch
  - Issue, bug
- 



# Flow

- Create a branch.
- Add your changes.
- Push.
- Create a Pull Request.
- Merges and Discussions.

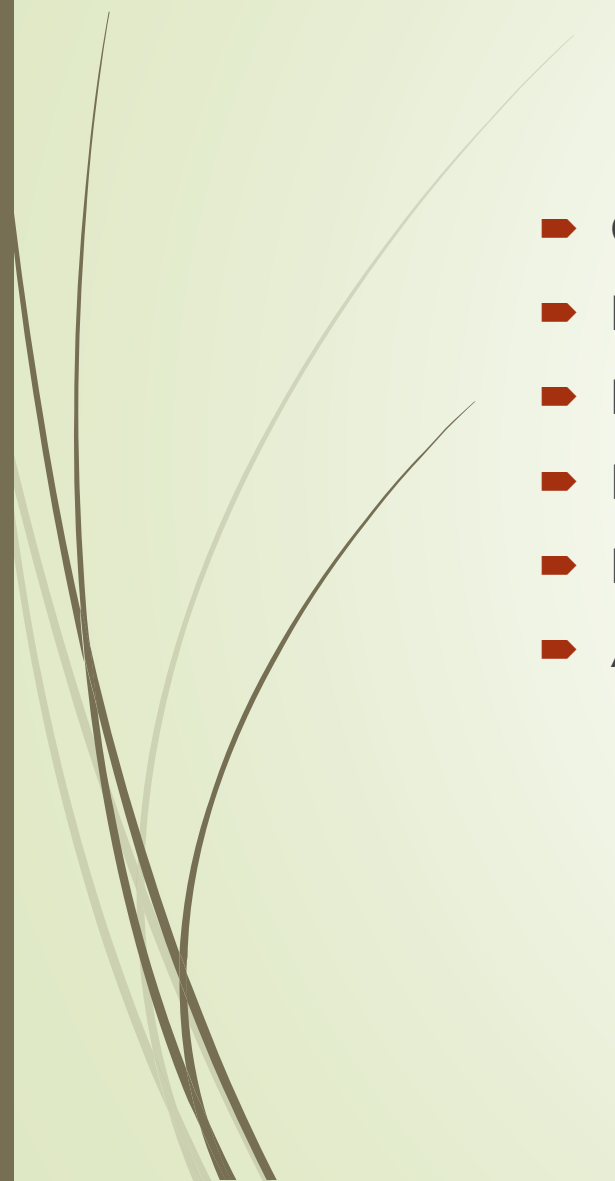


# Common Questions & Their Answers

- Is coding the only way?
- How to submit a good GitHub Pull Request?
- Open sourcing your own project is also an open source contribution.
- What organizations do to involve and welcome you?
- Summer and winter programs to get you in.
- Is everything on GitHub?
- Your Questions? Anything!

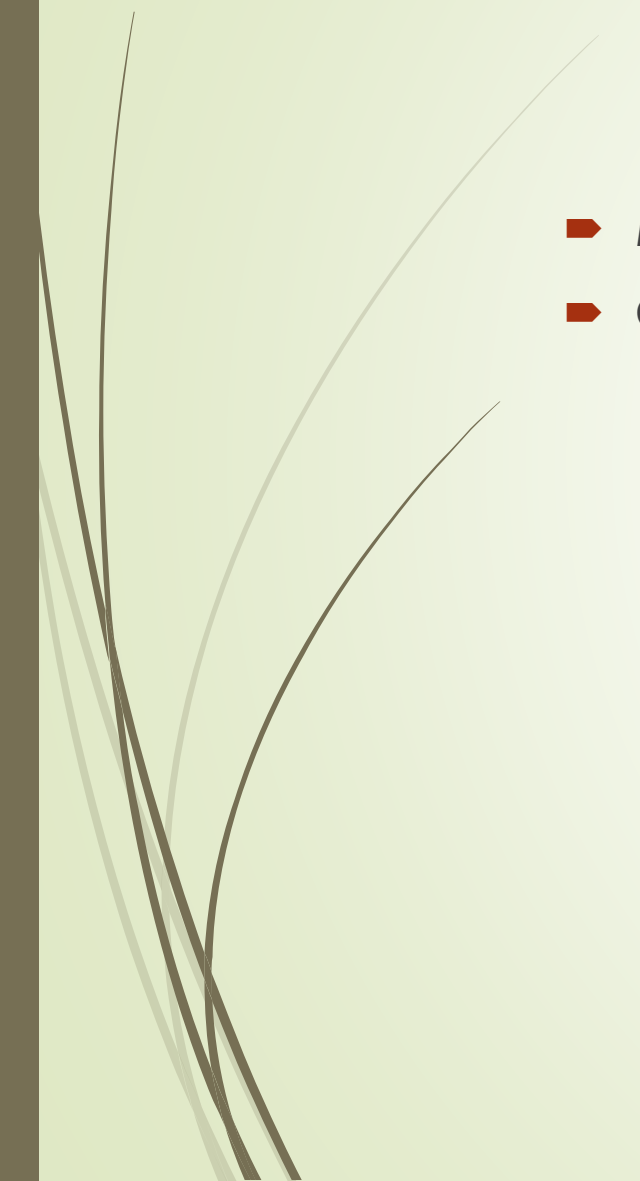


# Your contributions can be ..

- Code
  - Documentation Patches
  - Finding a bug
  - Raising an installation issue
  - Blogs about your experience
  - Answering a question
- 



# You can contribute by..


- Making your own project open source.
  - Contributing to an already existing project.
- 

# How does a good GitHub PR looks like?

Show event dates for job create and edit #191 Edit

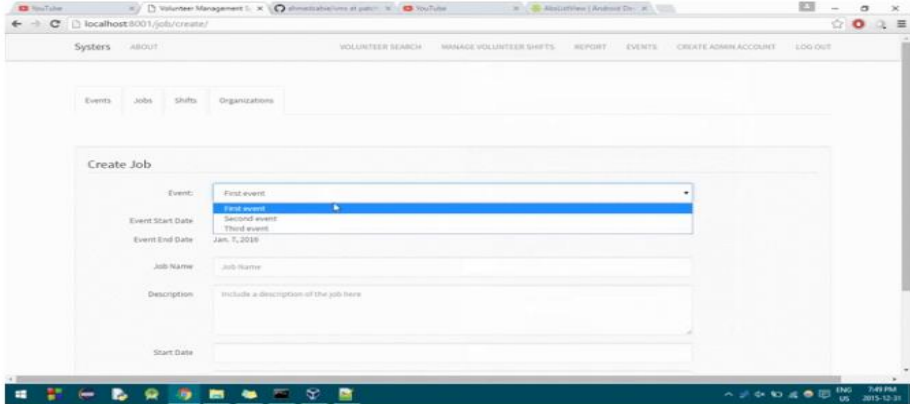
**Merged** vubo merged 1 commit into `systems:develop` from `ahmedsabee:patch6` on Jan 7

Conversation 7 Commits 1 Files changed 3 +40 -13

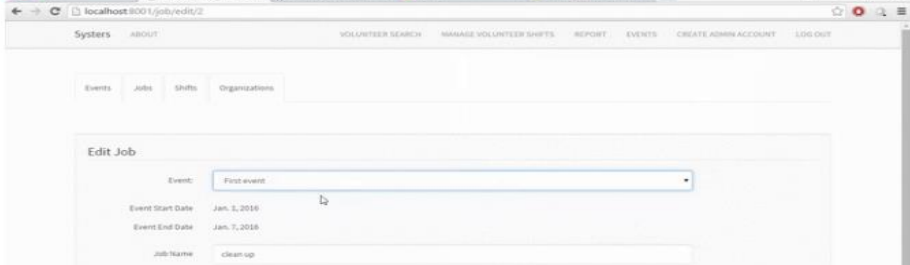
 ahmedsabee commented on Jan 3

The event dates are now shown in both job creation and editing as separate fields, and will change based on what event is chosen.

job creation example:



job editing example:




**Labels** ⚙  
None yet


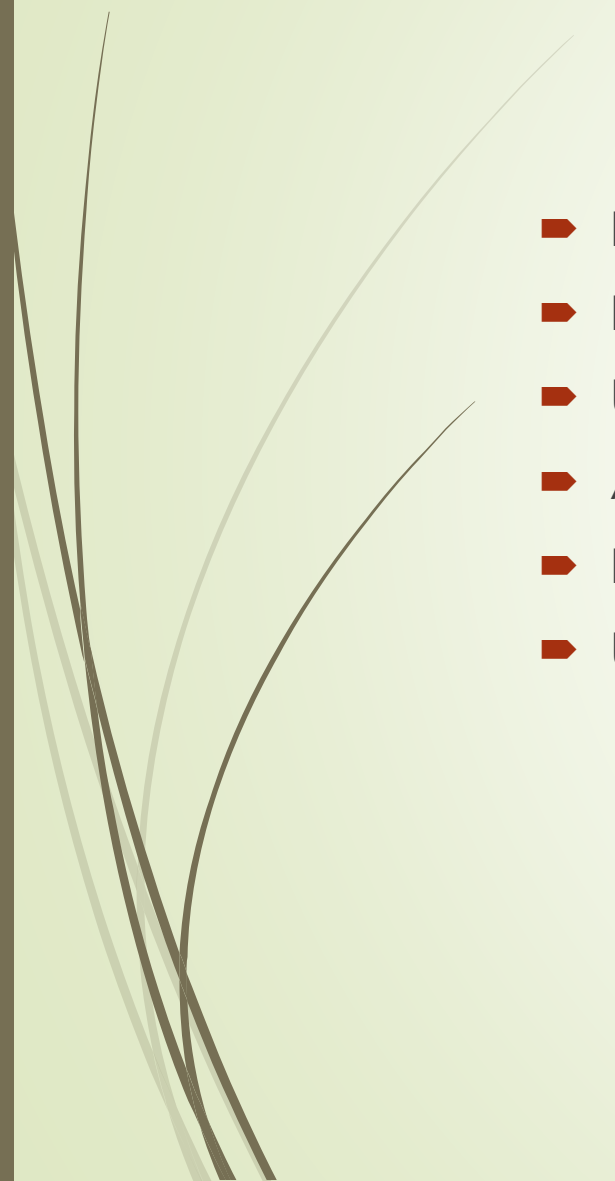
**Milestone** ⚙  
No milestone

**Assignee** ⚙  
No one—assign yourself

**Notifications**  
🔊 Unsubscribe  
You're receiving notifications because you were mentioned.

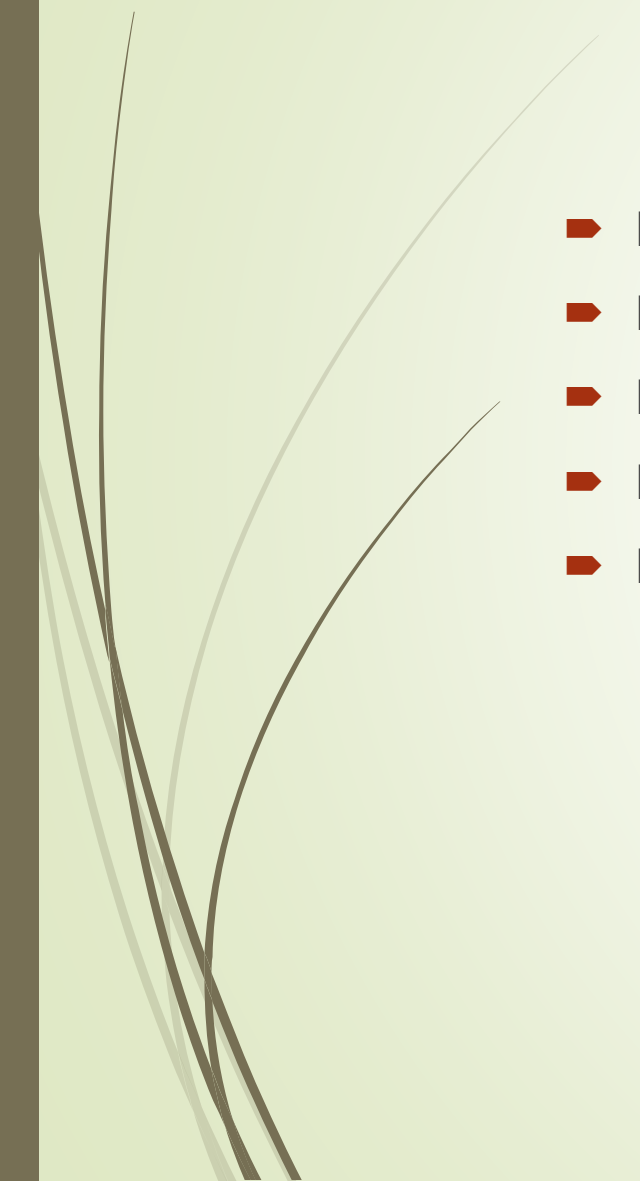
**3 participants**  


🔒 Lock conversation

- 
- 
- Proper commit messages.
  - Proper commit message division.
  - Using closes, fixes in your PR to reference the issue.
  - Adding a GIF or image of the output your code edits bring.
  - Explain your changes.
  - Use proper Markdown.



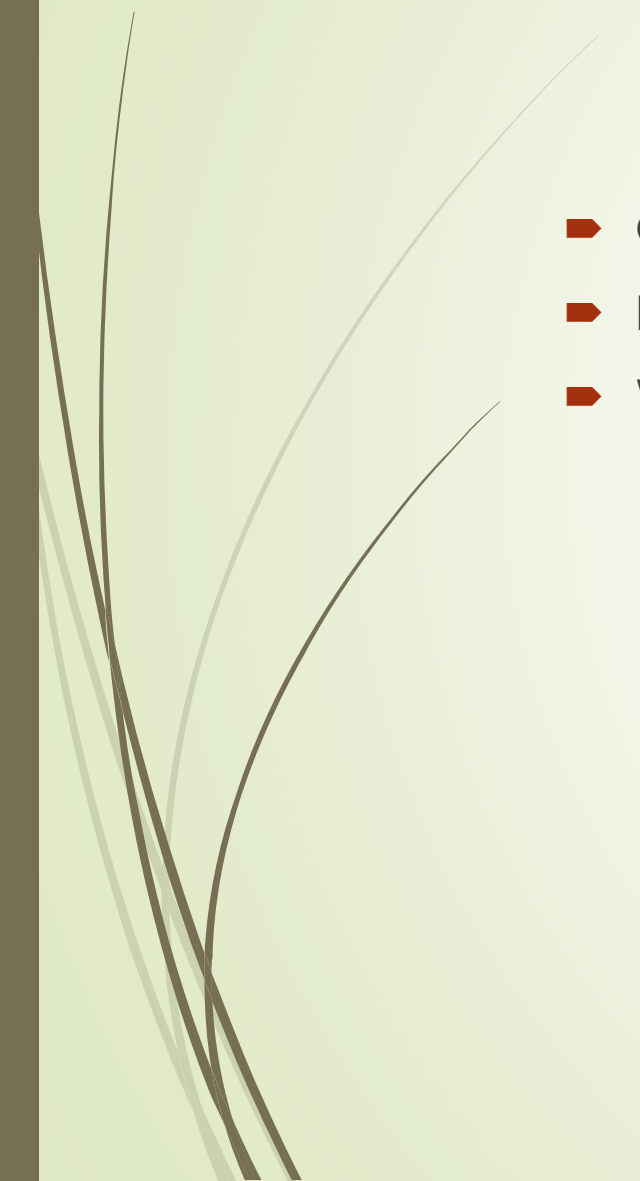
# Where to start?

- Find the source code, clone, install, run.
  - Find the documentation.
  - Find the issue/bug tracker. Pick newbie friendly bugs.
  - Find the mentor, maintainers of the project.
  - Find the IRC channel, mailing list.
- 





# Beginner Friendly Resources

- ▶ Gnome-love bugs.
  - ▶ Linux Kernel newbies.
  - ▶ WhatcanIdofoforMozilla?
- 



# Open Source Programs



- [Season of KDE](#)
- [OWASP SOC](#)
- [Mozilla Winter of Security](#)
- [Outreachy](#)
- [RGSOC](#)
- [TOR Summer of Privacy](#)
- [GSOC](#)
- [SOCIS](#)
- [More SOC-Programs](#)



# Are all open source project on GitHub?

No.





# Do's

- Read documentation.
- Ask.
- Learn if you don't know the technology that the project uses.
- Lurk on IRC's.
- Read.



# Don'ts

- Don't get scared, get started!
- 



Questions?



# References



- <http://semver.org/>
- <http://www.thegeekstuff.com/2010/09/linux-file-system-structure/>
- <https://guides.github.com/introduction/flow/>



Thank you.