Software Carpentry

Poruri Sai Rahul,
Software Developer, Enthought.
@rahulporuri,

- Version Control
- Project organization
- Software practices
- Documentation
- Collaboration
- Data management
- Misc

Software Carpentry

- Version Control
- Project organization
- Software practices
- Documentation
- Collaboration
- Data management
- Misc

- Version Control
- Project organization
- Software practices
- Documentation
- Collaboration
- Data management
- Misc

Version Control

- Avoid emails with attachments and file names with `-final` in them.
- Use Git/Hg/SVN for version control.
- Use GitHub/GitLab/BitBucket for collaboration and backup.
- Commit frequently and use PRs to review and accept changes.
- Maintain a CHANGELOG file, if not for every commit, for every major update to the codebase.

logo.svg

logo-2.svg

logo-3-monica-feedback.svg

logo-3-FINAL.svg

logo-3-FINAL-1.svg

By saving copies

`logo.svg`

By making commits

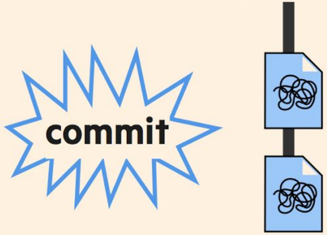
`logo.svg`

By saving copies

logo.svg

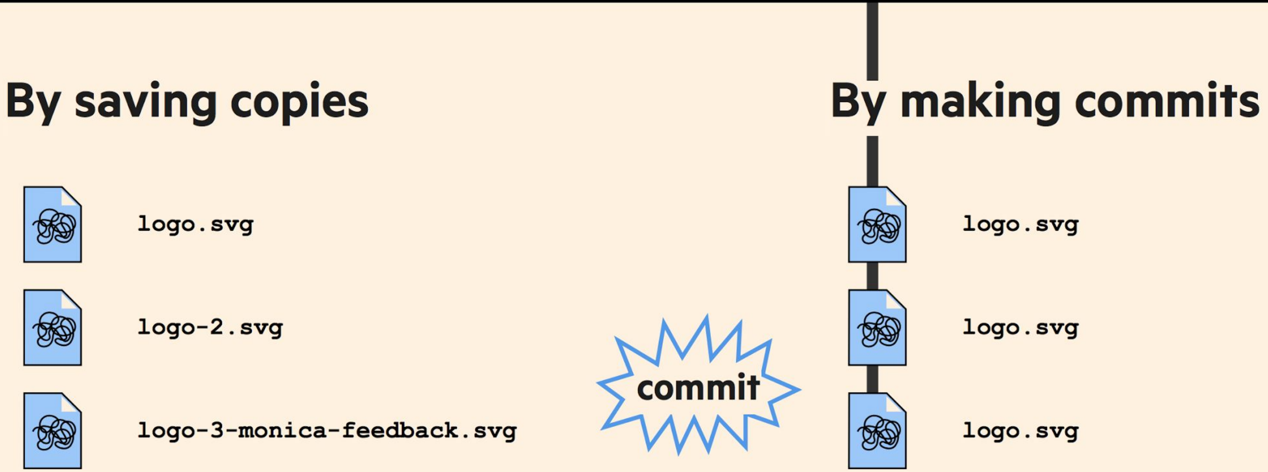
logo-2.svg

By making commits



logo.svg

logo.svg



By saving copies

logo.svg

logo-2.svg

logo-3-monica-feedback.svg

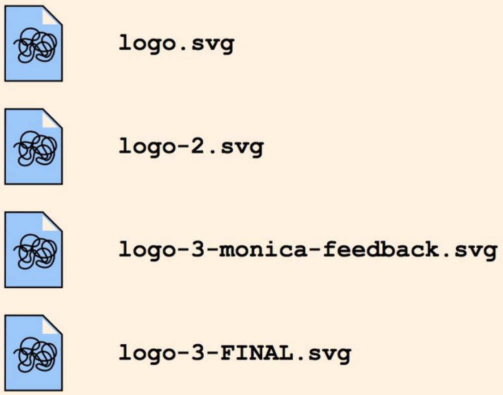
By making commits

logo.svg

logo.svg

logo.svg

By saving copies



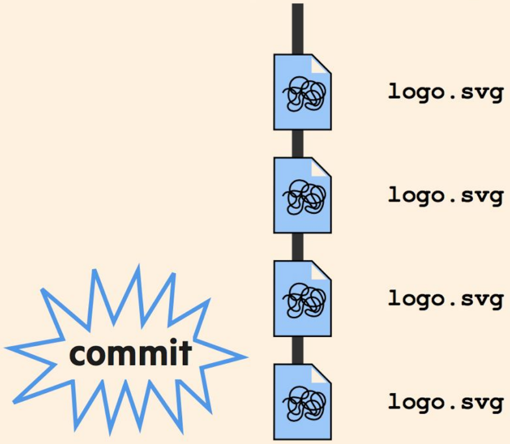
logo.svg

logo-2.svg

logo-3-monica-feedback.svg

logo-3-FINAL.svg

By making commits



logo.svg

logo.svg

logo.svg

logo.svg

By saving copies

`logo.svg`

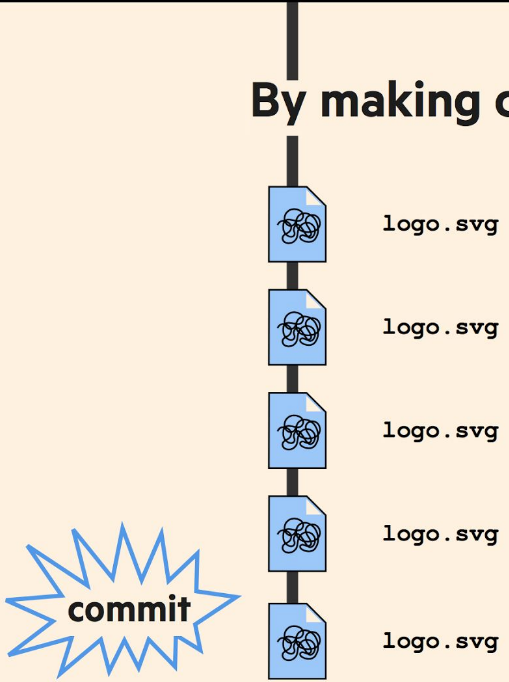
`logo-2.svg`

`logo-3-monica-feedback.svg`

`logo-3-FINAL.svg`

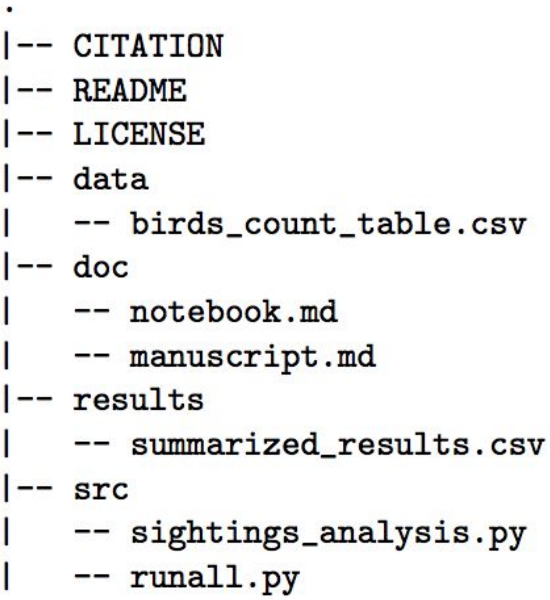
`logo-3-FINAL-1.svg`

By making commits



- Version Control
- Project organization
- Software practices
- Documentation
- Collaboration
- Data management
- Misc

Project organization



```
.
|-- CITATION
|-- README
|-- LICENSE
|-- data
|    -- birds_count_table.csv
|-- doc
|    -- notebook.md
|    -- manuscript.md
|-- results
|    -- summarized_results.csv
|-- src
|    -- sightings_analysis.py
|    -- runall.py
```

- Version Control
- Project organization
- Software practices
- Documentation
- Collaboration
- Data management
- Misc

Software practices

- Use local (function or class) scopes instead of global scopes - this will help the person that has to learn and change your code.
- Consistent and meaningful names - clarity in variable names will prevent the need for memorization.
- Consistent style and formatting - you can call it OCD but i'm going to call it laziness.
- Requirements file - please don't ask someone to figure out all the required packages by running and installing libraries incrementally.
- Dont leave commented code lying around - Why use Git if you are still going to leave around dead/useless code.

- Version Control
- Project organization
- Software practices
- Documentation
- Collaboration
- Data management
- Misc

Software practices

- Premature optimization - use profiling after code works.
- Write automated tests - you won’t like it when your code stops with an Error message after a full day of running the code.
- Learn how to use a debugger - stepping through the code is an incredible way to learn how the code actually works.
- Don’t rewrite. Reuse. Leverage open source libraries - stand on the shoulder of giants who have come before you.
- Automate workflow - a single command to take raw data and generate processed output and relevant plots.

- Version Control
- Project organization
- Software practices
- Documentation
- Collaboration
- Data management
- Misc

Documentation

- A README file, that contains any and all basic information on the code.
- A brief introduction on what the code does and an example use case.
- Document design and purpose, not mechanics or implementation
    - `i+=1 # Increment the variable i by one`
- Add docstrings, with example use cases, in functions/classes/methods.
- Going further, auto generate docs using Sphinx.

- Version Control
- Project organization
- Software practices
- Documentation
- Collaboration
- Data management
- Misc

Collaboration

- Use issue trackers on GitHub to keep track of bugs in the code or changes/features/enhancements that are pending.
- Code reviews - review each other’s code before merging them to your code base.
- Pair programming - two work faster and better than one.
- Add a LICENSE file (please don’t use GPL)
- CONTRIBUTING file - set the guidelines on how newcomers can contribute to your project.
- CITATION file - provide information on how people can cite your code in their work.

- Version Control
- Project organization
- Software practices
- Documentation
- Collaboration
- Data management
- Misc

Data management

- Save raw data. Never ever tamper with it. Never ever.
- Document all steps from raw data to processed data, plots - please don’t expect others to read your mind.
- Store data in a format that is easy to understand and analyze.

- Version Control
- Project organization
- Software practices
- Documentation
- Collaboration
- Data management
- Misc

Misc

- Automate repetitive tasks
- Use the right tools
  - use IPython for auto complete,
  - Use a modern editor for jump-to-def and other awesomeness.
- Master the terminal
  - reverse-i-search.
- Make files
- Continuous Integration
- Coverage