Tutorial on coding conventions and version control

Date: Sep 29th 2025

Coding conventions

- Keeps code clean, consistent, easier to maintain
- Python
  https://peps.python.org/pep-0008/
- Java
  https://google.github.io/styleguide/javaguide.html

  https://www.oracle.com/java/technologies/javase/codeconventions-contents.html

Version control

- Version control keeps file history and changes
- Allows separation of features
- Separation of concurrent work (e.g., branches OR forks)

Github

- https://git-scm.com/video/what-is-version-control
- https://git-scm.com/video/what-is-git
- https://graphite.dev/guides/git-fork-vs-branch

Working on a Fork

- A fork is a copy of a repo (usually on your account)

Basic steps

- **Fork** the main repo
- Make your changes
- Commit and push to your fork
- Send a **Pull Request** (PR) to the main repo
- PR is reviewed and merged

How to use forks / pull requests

- How to make a fork

https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/working-with-forks/fork-a-repo

- How to make a Pull Request (PR)

https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/proposing-changes-to-your-work-with-pull-requests/creating-a-pull-request

Git branches

- Creating a new branch

```
git checkout –b <your-branch-name>
```

Branching steps

- Make your **branch from master**
- Make changes on your branch
- Commit and push to your upstream branch
- Send a **PR** to merge your branch to master

- **How to make a pull request**

What is code review?

- We review each other's code after a feature/bugfix is completed
- Happens before merging code to master
- E.g., review together at desk, or on github PR

Goals

- Catch bugs early
- To keep code consistent
- Teamwork / learning from each other

Code review resources

- MIT code review guide

https://web.mit.edu/6.005/www/fa15/classes/04-code-review/

- Examples

https://www.qodo.ai/blog/java-code-review-checklist/

- Code review vs automated tests
- https://graphite.dev/guides/code-review-vs-automated-testing

Code review checklist

Basics

- Does the code do what it says it does?
- Check the code compiles
- Check old unit tests still pass

Testing

- Is the code unit tested?
- Are there any Exceptions that need to be handled?

Style

- Naming conventions (variables, class names)
- Is the code easy to read?
- Is the function documented (comments)?

Code review guideline

Person who wrote the code

- Describe the problem / feature
- Short summary of your solution (keep this concise)
- You (or I) can nominate code reviewers

Person reviewing the code

- One thing you liked / learned from the code
- One thing you see could be improved (or coding checklist)