Practical Git and GitHub Tips and Tricks for Beginners

Git enables powerful version control for projects, while GitHub expands its capabilities with features like issue tracking, pull requests, and project boards. However, Git and GitHub can seem daunting for beginners at first.

In this guide, we'll demystify Git and GitHub by covering practical tips and tricks for getting started:

- Using essential Git commands for staging, committing, branching, and pushing code changes.
- Undoing changes in Git with checkout, reset and revert.
- Tracking issues and task lists on GitHub.
- Building project boards with GitHub's Kanban workflows.
- Collaborating on code with contributors through pull requests and reviews.
- Avoiding common Git mistakes like leaking credentials.

Understanding these real-world Git and GitHub workflows will level up your version control skills and enable better collaboration. Let's dive in!

Handy Git Commands

Here are some common Git commands beginners should know:

- git init - Create a new Git repository in the current directory.
- git status - View any pending changes to files that need to be staged and committed.
- git add - Stage file changes for the next commit.
- git commit - Commit staged changes along with a descriptive commit message.
- git push - Push local commits to the remote repository like GitHub.

Undoing Changes in Git

Mistakes happen, but thankfully there are a few ways to undo changes in Git:

- git checkout -- <file> - Discard unstaged changes in the given file.

- git reset - Reset the state back to a previous commit.
- git revert - Revert a committed change by creating a new commit.

GitHub Issues for Task Tracking

GitHub's issue tracker lets you create issues to capture bugs, tasks, feature requests, etc.

Useful tips:

- Assign issues to collaborators and set milestones to organize work.
- Label issues for quick filtering and reporting.
- Reference issues in commits and pull requests to close them.

GitHub Project Boards

GitHub Project boards provide Kanban-style task boards for managing workflows:

- Create columns like "ToDo", "In Progress", "Completed"
- Add cards representing tasks, issues, or features requests.
- Move cards across columns to visualize progress.

Collaborating with Git via Pull Requests

Pull requests enable collaborators to review code changes on GitHub before merging:

- Fork a repository and create a branch with your changes.
- Open a pull request to propose your branch's changes.
- Teammates can review, comment, and approve changes.
- Merge the pull request once reviews are passed.

By mastering these key Git and GitHub workflows, you'll boost your productivity and ability to collaborate on projects!

Avoiding Common Git Mistakes

Some key pitfalls to avoid as a Git beginner:

- Don’t commit large files or credentials - use .gitignore.
- Resolve merge conflicts carefully to avoid losing work.
- Understand Git branching models to maintain a clean history.

With these tips, you can avoid headaches down the road.

Adopting these practical workflows will help you become proficient with Git and GitHub. Version control and collaboration are essential skills for any developer - and getting comfortable with Git doesn’t have to be intimidating. What are you waiting for? Start leveling up your Git skills today!