# Git --- Let's talk about it

**Author(s)**: Louis Ouellet

---

First off, I'll be the first to admit: I'm no Git expert. I'm a beginner and still learning, but I wanted to write this article to share what I know so far and hopefully help others who are just starting out. I've really only been using Git consistently since last year, primarily as a backup solution for my projects. But I've come to realize that Git is far more than a simple backup tool. It's a powerful asset for managing projects and collaborating with others.

About a year ago, I learned the basics of commits and why they're so important. Since then, I've made an effort to improve my own commit messages, aiming to make each one as detailed as possible. Recently, I started exploring GitHub Workflows and Actions, discovering ways to automate aspects of my projects. So while I'm still learning, I'm excited to share what I know with you. Let's start with the basics.

## What is Git?

Git is a distributed version control system that lets you track changes in your code. Created by Linus Torvalds in 2005, Git has become an essential tool in the software development industry. It helps developers manage projects and work together, tracking every change in a reliable, structured way.

## Why you should use Git?

There are many reasons why you should use Git. Here's what I think are the most important

ones:

- It's a great tool to manage your projects.
- It lets you keep track of changes in your code over time.
- It makes collaboration with others easier.
- It’s surprisingly simple to get started.

## How to get started with Git?

To get started with Git, you need to install it on your computer. You can download Git from the official website. Once you have installed Git, you can start using it by creating a new repository. You can create a new repository by running the following command in your terminal:

```
git init
```

This will create a new repository in the current directory. You can then start tracking changes in your code by creating a new commit. You can create a new commit by running the following command in your terminal:

```
git add .
git commit -m "Your commit message"
```

This will create a new commit with all your changes. You can then push your changes to a remote repository by running the following command in your terminal:

```
git push
```

This will push your changes to a remote repository. You can then collaborate with others by sharing your repository with them.

## How can you improve your commits?

To improve your commits, you need to follow some best practices. Here are some tips to help you improve your commits:

- Write detailed commit messages.
- Use meaningful commit messages.
- Keep your commits small and focused.

One great tool is the patch mode. You can use the patch mode to review your changes before

committing them. You can enter the patch mode by running the following command in your terminal:

```
git add -p
```

or

```
git add --patch
```

This will enter the patch mode and allow you to review your changes before committing them. You can then select which changes you want to commit and which changes you want to discard. That way your commits will be more focused and easier to review.

## Collaborating with Team Members

To collaborate with other team members, you need to share your repository with them. You can share your repository by adding them as a collaborator. You can add a collaborator by running the following command in your terminal:

```
git remote add origin git@github.com:User/UserRepo.git
```

This will add a remote repository to your local repository. Each team member can now create a new branch and start working on their changes. They can then push their changes to the remote repository by running the following command in their terminal:

```
git push origin branch-name
```

Once a team member wants to merge their changes with the main branch, they can create a pull request. They can create a pull request by going on the GitHub website and clicking on the “New pull request” button. They can then review their changes and merge them with the main branch.

## What if I want to contribute to someone else's project?

If you want to contribute to someone else's project, you need to fork their repository. You can fork a repository by clicking on the “Fork” button on the GitHub website. Once you have forked the repository, you can clone it to your local machine and start working on your changes. You can then push your changes to your forked repository and create a pull request to the original repository.

## Isn’t This a Lot of Extra Steps?

Yes, there are quite a few steps, but they’re worth it for the control and clarity Git provides. Plus, GitHub offers tools like Workflows and Actions to automate some of these processes. For example, you can create a workflow that runs tests and builds your project every time you push changes. You can even automate pull request merges after approvals, saving time and boosting efficiency.

## Conclusion

Git is a powerful tool that can help you manage your projects and collaborate with others. It's easy to get started with Git and there are many resources available to help you learn. I hope this article has helped you understand the basics of Git and how you can use it to improve your projects. If you have any questions or need help, feel free to ask. I am always happy to help. Happy coding!

## Tagsgeneralgittoolscollaboration

- Twitter
- Facebook
- LinkedIn
- Reddit
- Telegram
- Email