Poznan University of Technology

Git - introduction to version control

Jakub Piotr Hamerliński, M.Eng.

Version control

Agenda

- Git
- GitHub
- GitHub CLI

Git

Version control

Git

Git is a version control system that allows you to track changes to your code over time and collaborate with other developers. It helps you keep track of all the changes you make to your code, as well as the changes made by your collaborators.

Version control

Git basic operations

There are a few basic operations in Git that you should be familiar with:

- **git clone**: This command is used to create a local copy of a repository that is hosted on a remote server, such as GitHub.
- **git add**: This command is used to add changes that you have made to your local repository to the staging area. This means that the changes are ready to be committed, but have not yet been permanently saved to the repository.
- **git commit**: This command is used to save changes that have been added to the staging area to the repository. When you commit changes, you should provide a message explaining what changes you made and why.

Version control

Git basic operations

- **git push**: This command is used to send your committed changes to a remote repository, such as GitHub.
- **git pull**: This command is used to fetch and merge changes from a remote repository into your local repository. These are just a few of the basic operations in Git, but there are many more commands and features available. It's a good idea to familiarize yourself with the basics of Git before diving into more advanced features and workflows

GitHub

Version control

GitHub

GitHub is a web-based platform that is used for version control and collaboration on software projects. It allows developers to work on code together, track changes to the codebase, and roll back changes if necessary. It also provides a platform for users to host and share their software projects with others.

Git is the version control system that GitHub is built on top of. Git allows developers to track changes to a codebase over time, revert back to previous versions of the code, and collaborate with other developers on the same codebase. GitHub provides a user-friendly interface for developers to use Git, as well as a range of tools and features to make collaboration and project management easier.

GitHub is widely used by developers and organizations around the world to host and collaborate on software projects. It is also a popular resource for finding and sharing open-source software projects.

Version control

Example GitHub task

- Create a new repository on GitHub.
- Clone the repository to your local machine using the **git clone** command.
- Create a new file in the repository and add some content to it.
- Use **git add** and **git commit** to add the file to the repository and commit your changes.
- Push the changes to the remote repository using **git push**.

Version control

Example GitHub task

- Create a new branch in the repository using **git branch**.
- Checkout the new branch using **git checkout**.
- Make some changes to a file in the new branch and commit the changes.
- Push the changes to the remote repository using **git push**.
- Create a pull request to merge the changes from the new branch into the main branch.

GitHub CLI

Version control
GitHub CLI

GitHub CLI is a command line interface (CLI) tool that allows users to interact with the GitHub service and perform various actions directly from their terminal or command prompt. With GitHub CLI, users can create, clone, and fork repositories, create and edit issues and pull requests, and manage their git repositories and branches. It also provides additional functionality such as the ability to search for repositories and users, and to manage and configure various aspects of their GitHub account. GitHub CLI is designed to be a faster and more efficient way to work with GitHub, and can be used in conjunction with other Git tools or as a standalone tool.

Version control
GitHub CLI installation

To install GitHub CLI, you will need to have a recent version of Git and Node.js installed on your system. Once these dependencies are installed, you can follow these steps:

- Open your terminal or command prompt and enter the following command: **npm install -g gh**

This will install the GitHub CLI tool globally on your system.

- Once the installation is complete, you can verify that the GitHub CLI is installed and available by running the following command: **gh --version**

This should print the version number of the GitHub CLI to your terminal.

Version control

GitHub CLI installation

- If you want to configure the GitHub CLI with your GitHub account, you can run the following command:

  gh auth login

This will open a web browser and prompt you to log in to your GitHub account. Once you have logged in, you will be asked to authorize the GitHub CLI to access your account.

- Finally, you can test the GitHub CLI by running a simple command such as: **gh repo list**

This should list all of the repositories in your GitHub account.

Version control

Example GitHub CLI task

- Create a new repository on GitHub using the GitHub CLI. To do this, open a terminal and type:
- **gh repo create** *repo_name*
- Replace *repo_name* with the desired name for the repository.
- Clone the repository to your local machine using the GitHub CLI. To do this, type:
- **gh repo clone** *repo_name*
- Replace *repo_name* with the name of the repository you just created.
- Create a new file in the repository using the GitHub CLI. To do this, navigate to the local repository on your machine and type:
- **gh repo create-file --message "feat: add new file" --content "Hello, world!"** *file_name*
- Replace *file_name* with the desired name for the new file.
- Push the new file to the remote repository using the GitHub CLI. To do this, type:
- **gh repo push**

Version control

Example GitHub CLI task

- Check the status of the repository using the GitHub CLI. To do this, type:
- **gh repo status**
- Check the history of the repository using the GitHub CLI. To do this, type:
- **gh repo history**
- Check the list of collaborators on the repository using the GitHub CLI. To do this, type:
- **gh repo list-collaborators**
- Add a collaborator to the repository using the GitHub CLI. To do this, type:
- **gh repo add-collaborator** *username*
- Replace *username* with the desired username of the collaborator.
- Remove a collaborator from the repository using the GitHub CLI. To do this, type:
- **gh repo remove-collaborator** *username*
- Replace *username* with the username of the collaborator you want to remove.