VŠB TECHNICKÁ UNIVERZITA OSTRAVA | FAKULTA ELEKTROTECHNIKY A INFORMATIKY | KATEDRA INFORMATIKY

Basics of Information Technology

Ing. Michal Radecký, Ph.D. MBA

Versioning systems, GIT

What are versioning systems

- Versioning systems are tools that are used to track and store changes to code, documents, or any files during project development.
- These systems allow developers and other team members to collaborate on "the same" data.

Benefits of versioning systems

- Tracking changes and recording history
- Renewal within history
- Collaboration in creation (multi-user approach)
- Parallel development
- Reduce the risk of errors and conflicts

Types of versioning systems

Local

- They work on only one computer.
- Each developer maintains their own copy of the project.
- RCS (Revision Control System)

Centralized

- All files and their history are stored on a central server.
- Developers work with files that they download and upload (share with the server).
- One critical point in the event of an outage, etc.
- CVS (Concurrent Versions System), Subversion (SVN)

Decentralized

- Each developer has a complete copy of the repository, including the history.
- It allows parallel development.
- Git

Hlavní funkce

Commit
Saving changes to a repository with a short description of the changes made.

Branches
Create branches for parallel development without affecting the main branch.

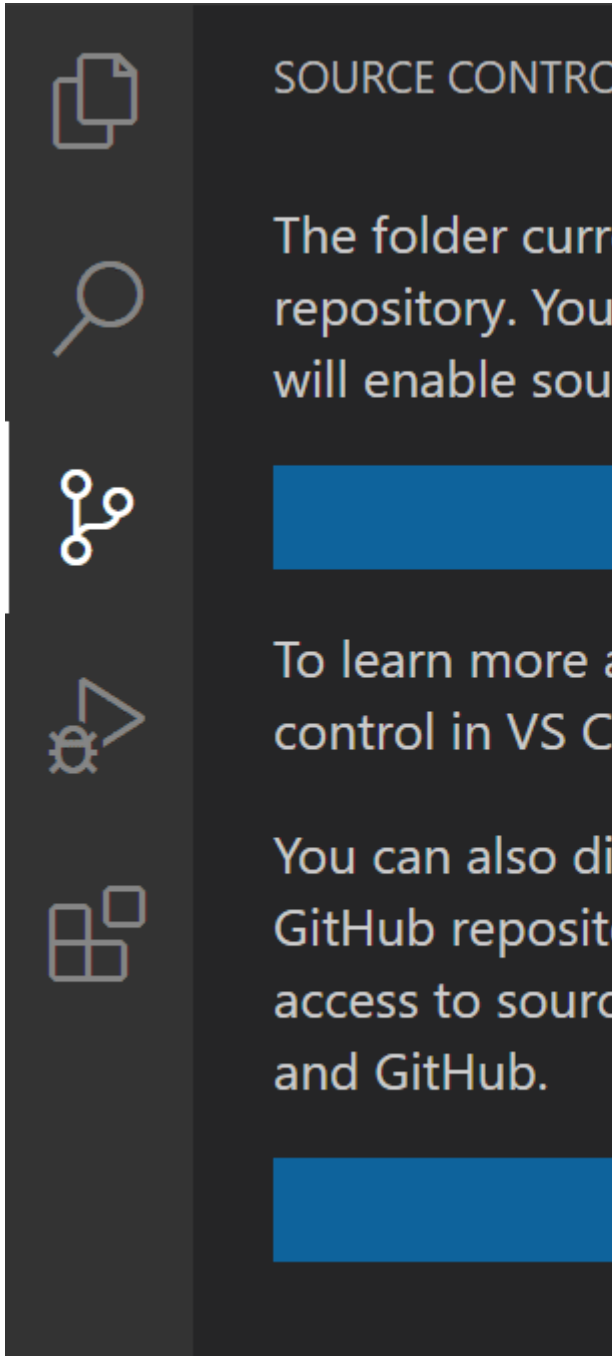
Merge
Unify changes from one branch to another.

Pull/Push
Update the local repository with changes from the central repository or vice versa.

Clone
Transfer/clone repositories – create a local one in a link to a remote repository.

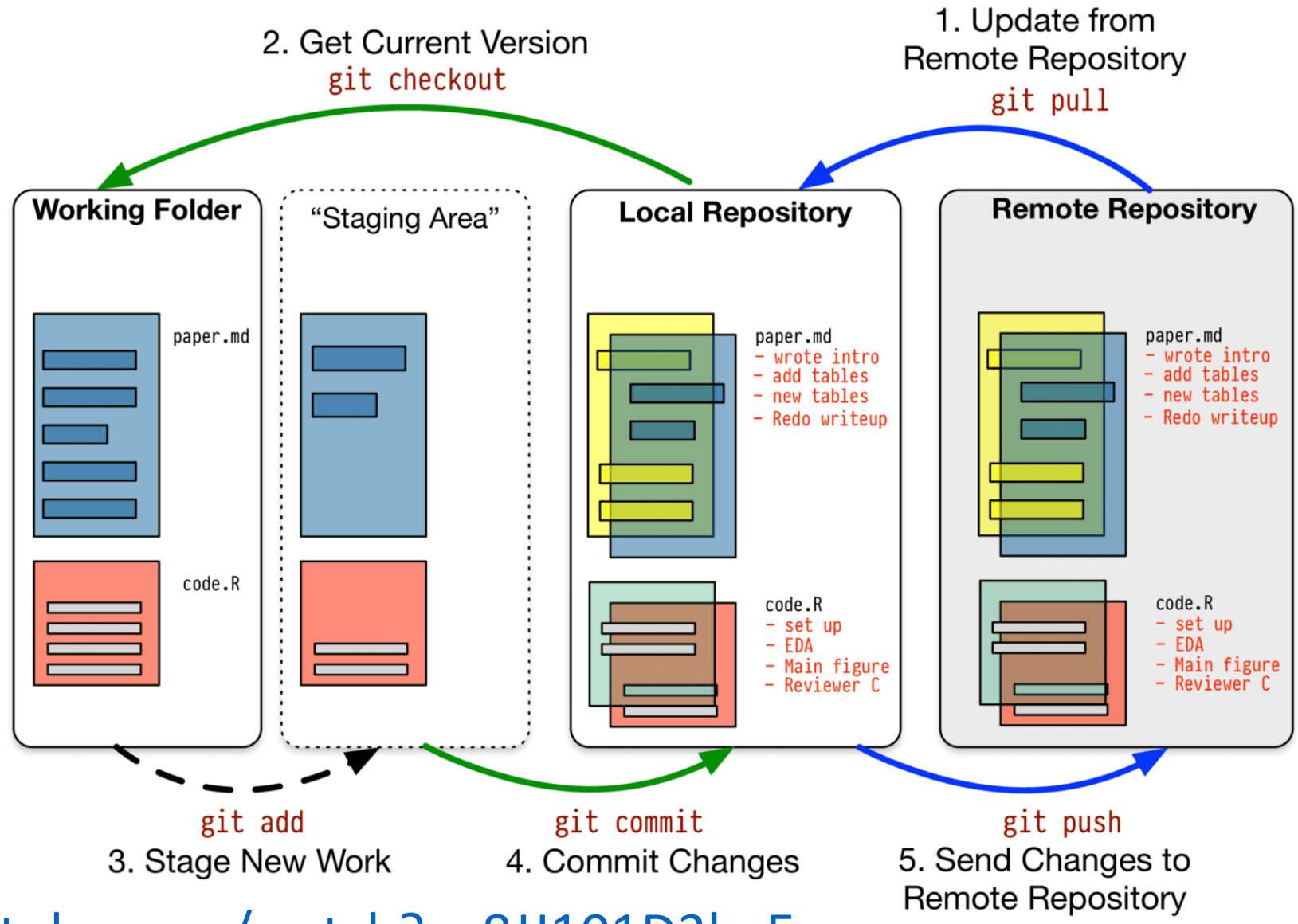
Git

- Git is a distributed/decentralized versioning system used for tracking code changes and versioning files during software development.
- It was developed by Linus Torvalds in 2005 and has since become one of the most popular instruments of this type.
- Git is open source software and is free to use.
- It allows you to support collaborative work and easily share code through remote repositories (GitHub, GitLab, Bitbucket).
- Wide Support of Various Tools and IDEs.
  https://code.visualstudio.com/docs/sourcecontrol/overview



https://cloudstudio.com.au/2021/06/26/git-command/, https://chiamakaikeanyi.dev/working-with-git/

Main principle



https://www.youtube.com/watch?v=8JJ101D3knE

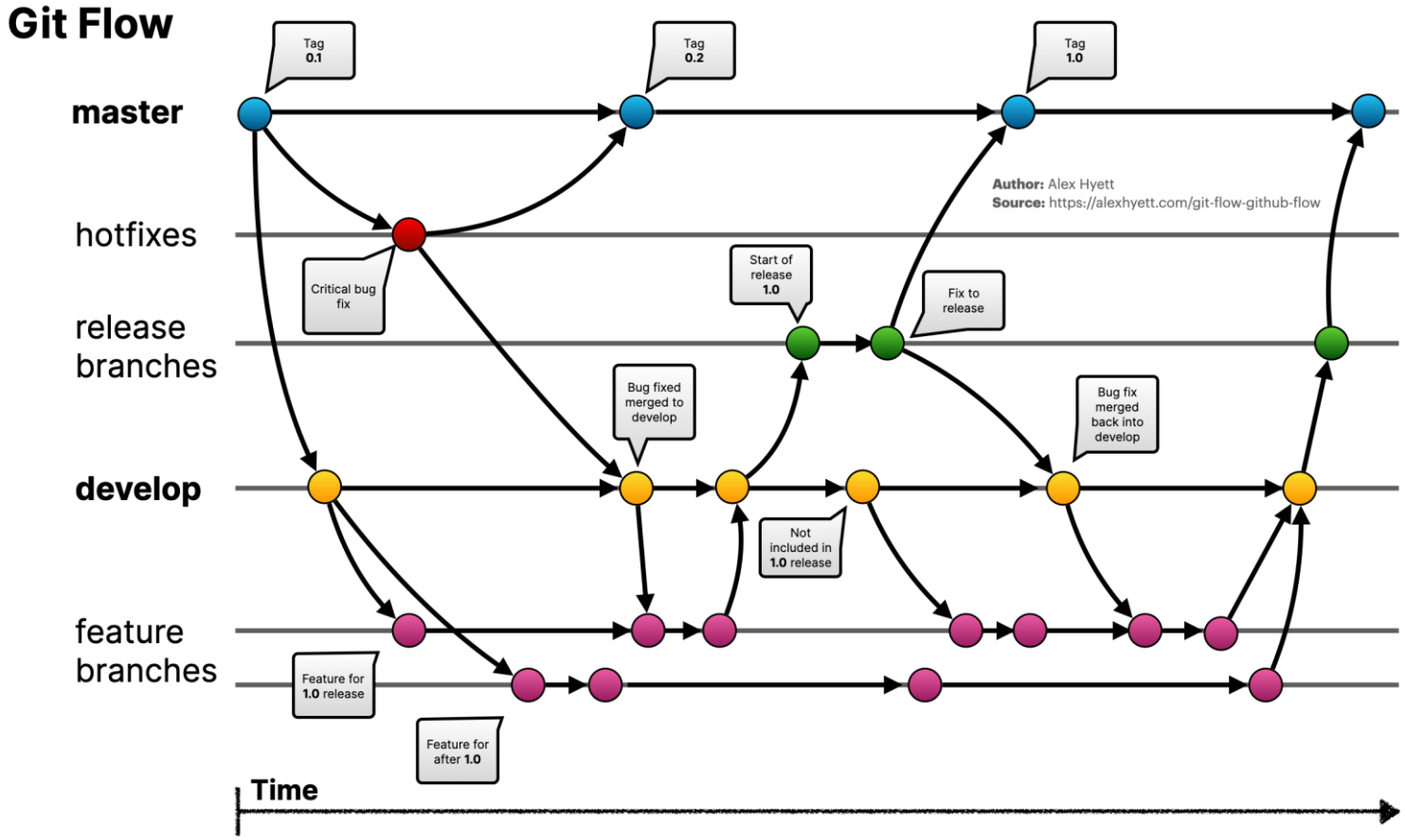
Staging Area

- Index/State Machine - a key concept that allows precise control of which changes will be included in the next commit.
- It acts as a staging space for the commit and the next stages of the process.
- Changes that are added here are flagged, but have not yet been checked out to the local repository itself.
- Tagging can be controlled by "add", which brings flexibility between the working version and the commit (working directory x staging area x commit)

https://www.alexhyett.com/git-flow-github-flow/

Git Flow

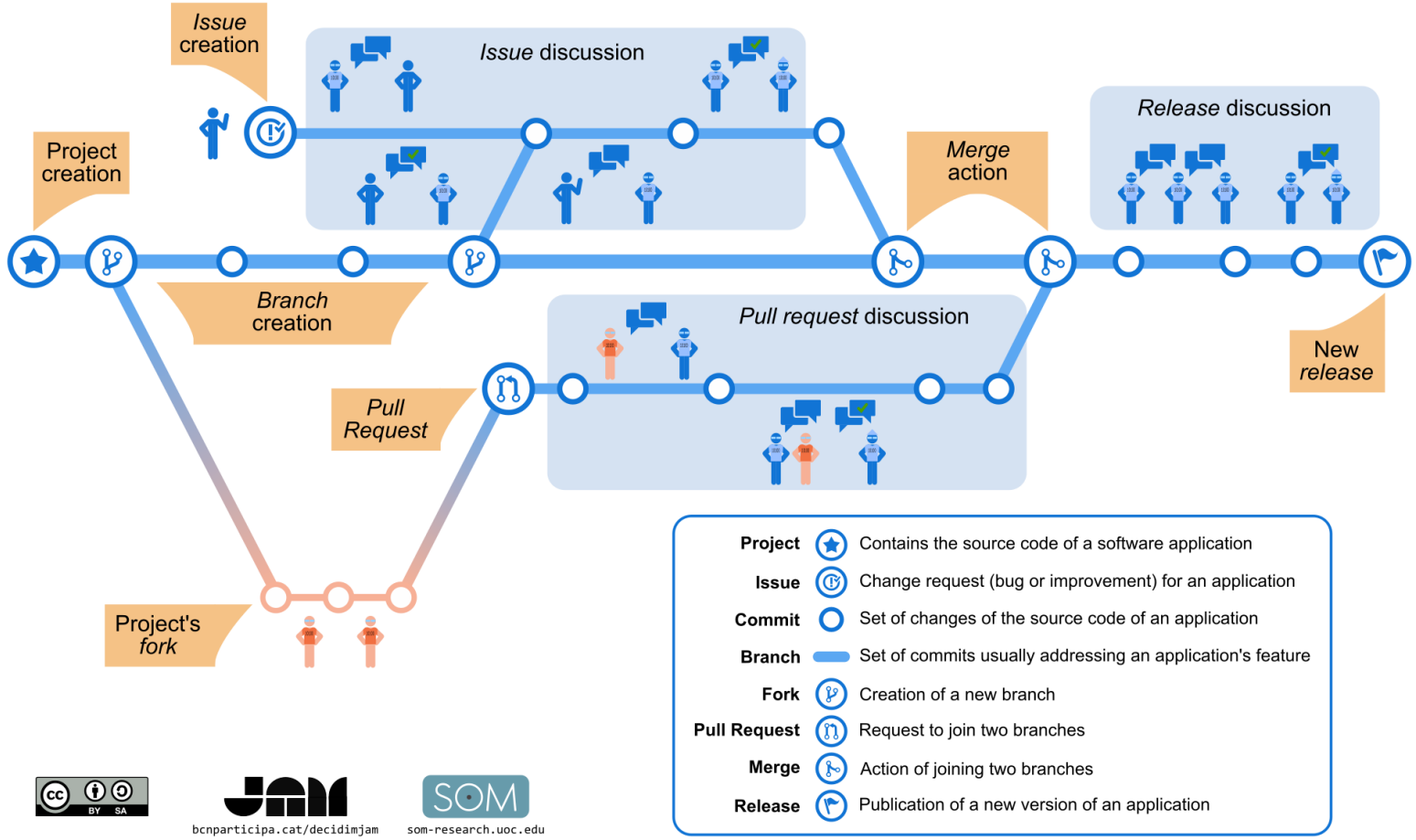
- Efficient development versionin
- Development history, documen



- **Branch** (master/main)
  The way it is distributed project into independent copies with isolated development
- **Merge**
  Connecting two branches
- **Fetch, Pull, Push**
  processes for synchronization between types of "repositories" with respect to defined branches, etc.

GitHub Flow

- Effective cooperation on development
- Review the code you create



- **Pull request**
  A requirement to merge changes from one branch to another, typically as part of teamwork
  After approval/review, a merge between branches is performed

Installation

- Git itself vs. visual "add-on tools"
  - Git BASH, Git GUI, GitKraken, TortoiseGit, …
- Installation
  - Windows - https://git-scm.com/download/win
  - macOS – `brew install git`
  - Linux – `sudo apt update`, `sudo apt install git`
- Configuration

```
git config --global user.name "Tvé Jméno„
git config --global user.email tvuj@email.com
```

-

```
git config –-list
git --version
```

https://sysxplore.substack.com/p/12-git-commands-everyone-should-know

Commands

git init

Initializes a new Git repository in the current directory.

```
$ mkdir railsapp
$ cd railsapp
$ git init
```

git add

Stages changes in the current directory and sub directories.

```
$ echo “# RAILS APP” > README.md
$ echo “Demo” >> README.md
$ git add README.md
```

git commit

Commits staged changes with a commit message.

```
$ git add app.css
$ git commit -m “Initial commit”
```

git push

Pushes local changes to a remote repository.

```
$ git add .
$ git commit -m "Update files"
$ git push origin main
```

git pull

Pulls changes from a remote repository and mergres into the local repository.

```
$ git status
$ git pull origin main
$ git log --oneline
```

git remote

Add a remote repository, view it, and rename it.

```
$ git remote add origin <url>
$ git remote -v
$ git remote rename origin upstr
```

https://sysxplore.substack.com/p/12-git-commands-everyone-should-know

Příkazy

`clone, diff, revert, log, switch, …`

git branch

List all branches, create a new branch with a specifed name, and confirm it's created.

```
$ git branch
$ git branch feature-login
$ git branch
```

git fetch

Retrieves the latest data from a remote repository - but it doesn't integrate any of this new data into your working files.

```
$ git fetch origin
```

git checkout

Switches to the specified branch.

```
$ git branch
$ git checkout feature-login
$ git branch
```

git merge

Merges the specified branch into the current branch (in this case main).

```
$ git checkout main
$ git merge feature-login
$ git log --oneline
```

git status

Displays the status of the repository, including any uncommitted changes

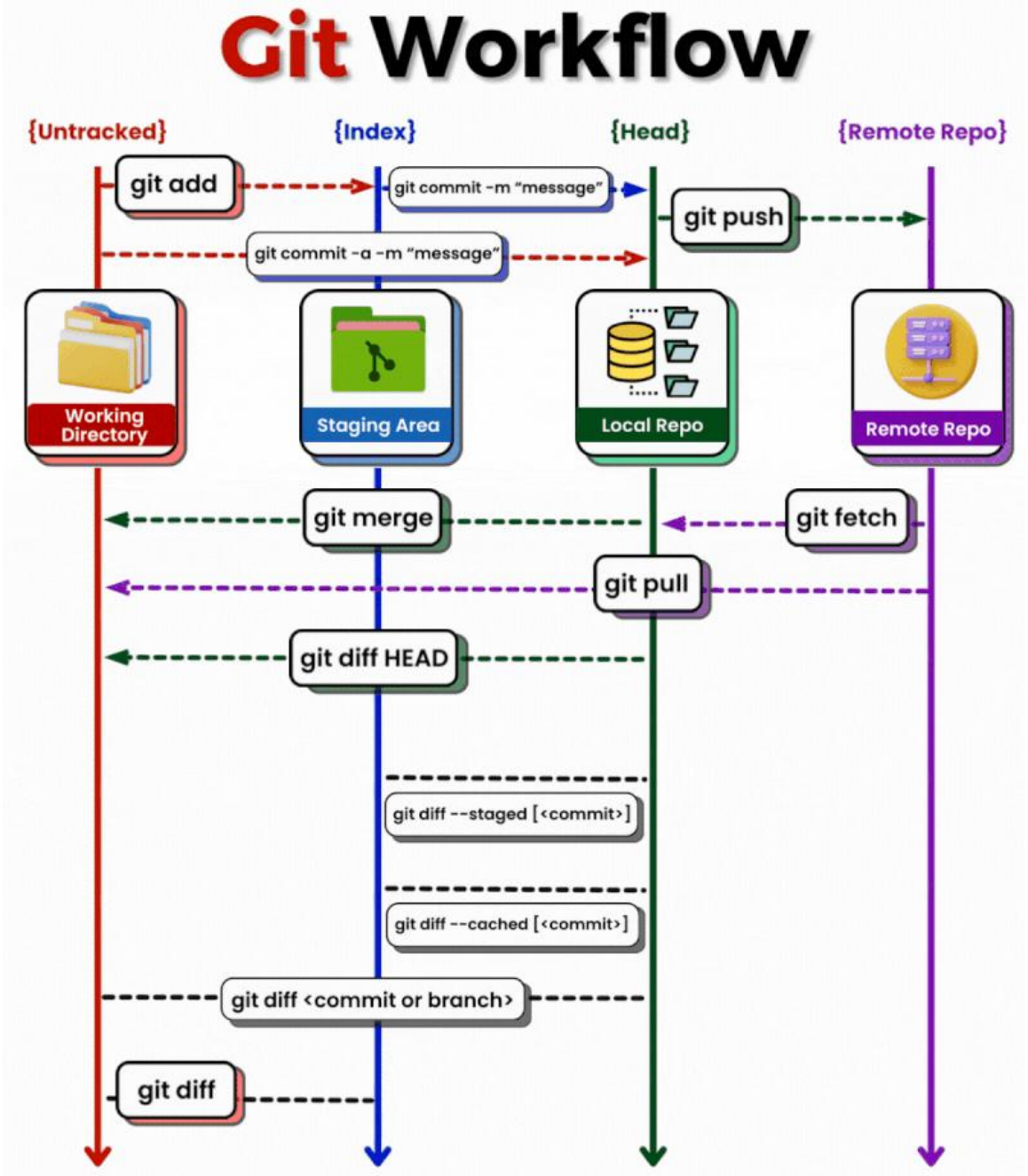
```
$ git status
$ echo "Data Added" >> hello.txt
$ git status
```

git reset

Resets the current branch to the specified commit

```
$ git log --oneline
$ git reset --hard <commit-hash>
$ git status
```

https://blog.devgenius.io/git-workflow-complete-guide-for-beginners-with-commands-examples-afcccd50f195



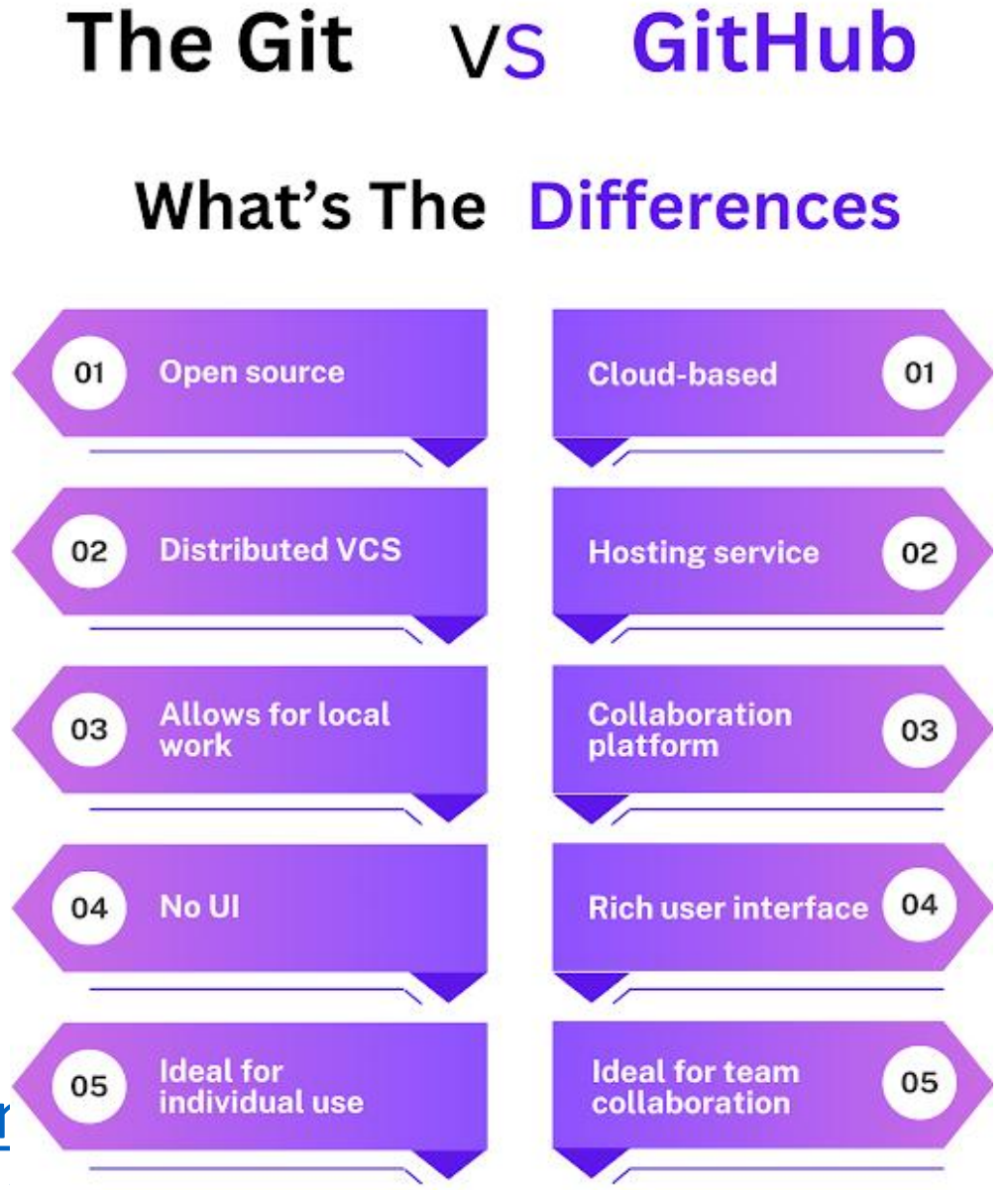
GitHub

GitHub is a web-based version control platform and hosting software projects using Git.

This is one of the most popular (420+ miles) Remote Repositories for developers, who want to share their code, collaborate on projects and track the history of changes.

Alternatives: GitLab, Bitbucket, Azure DevOps

https://github.blog/developer-skills/programmir frameworks/what-is-git-our-beginners-guide-to-



GitHub

Repositories
- Users can create repositories for their projects. The repository contains all project files and the change history.

Stars and Forks
- Users can evaluate or create their own projects/repositories from existing ones.

Issues a Pull Requests
- GitHub provides tools for tracking issues and pull requests. This allows for effective communication and collaboration within the team.

Collaborators and Teams
- Users can collaborate on projects by adding collaborators or creating teams with access rights.

Wikis and Projects
- Wikis allow you to create documentation and project pages. Projects offers tools for tracking tasks and organizing work.

Gists
- Allows users to share and discuss small pieces of code, text, or images.

GitHub Actions
- Provides tools for automating processes within the repository, such as build, test, and deployment.

GitHub Marketplace
- A platform for finding and integrating tools and applications into GitHub repositories.

Security and Code Scanning
- Tools for security code scanning and repository security.

Copilot
- Integration to leverage AI elements in project development and management.