Git Fundamentals

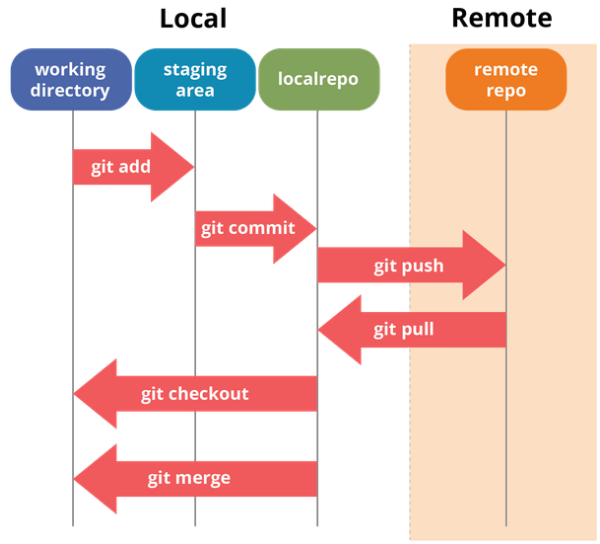
Version Control

- Git (Version control System)
- Used by anyone who creates content
    - Versions update
    - History of changes / commits
    - Revert to earlier version
- Offsite copy of data
    - Easy synchronisation with local copy
- Supports collaboration
    - People can update simultaneously
    - Overlapping changes
    - Branches - (dev) feature or experimental branches and merging to main
    - Merging and dealing with conflicts

Git

- CLI called **Git bash** to run commands
- Git server used to copy data - GitHub GitLab BitBucket
- Graphical UI - GitHub Desktop and IDE integration
- Directory or folder controlled by git is a repository or repo
- Single branch or multi branch
- Scope levels: System, Global, then Local
    - Change default branch name
    - **Git config --global init.default branch main**
- Commands (*pwd, mkdir GitExample, cd, ls or ls -al*)
    - Git init (run in folder to be controlled by git)
        - Creates a sub folder called .git, stores config information for the repo, remembers all history and versions of the files, all branches, keeps track of currently working HEAD
- **.gitignore t**emplates to exclude files from my repo such as .env or .properties
- HEAD - reference to the currently checked-out branch latest commit

Staging and committing files



Commands

- **Git add**
- **Git status** - shows untracked files
- **Git commit** -m “Initial commit”
- **Git diff** - difference between working files and commits
- **Git log**

Github

- Copy a local repo to Github
  - Copy the HTTPS URL
  - Set the connection to GitHub and push
  - **Git remote add origin <url>**
  - **Git push origin main**
  - Create personal access token
  - All tracked files will be copied to GitHub
- Clone a repository
  - **Git clone <url>**
- Pull a new version
  - Checks differences between local and remote repos and downloads changes and updates local files
  - **Git pull**