Git

Σέργιος - Ανέστης Κεφαλίδης
Κωνσταντίνος Νικολέτος
Κώστας Πλας

Git & Github

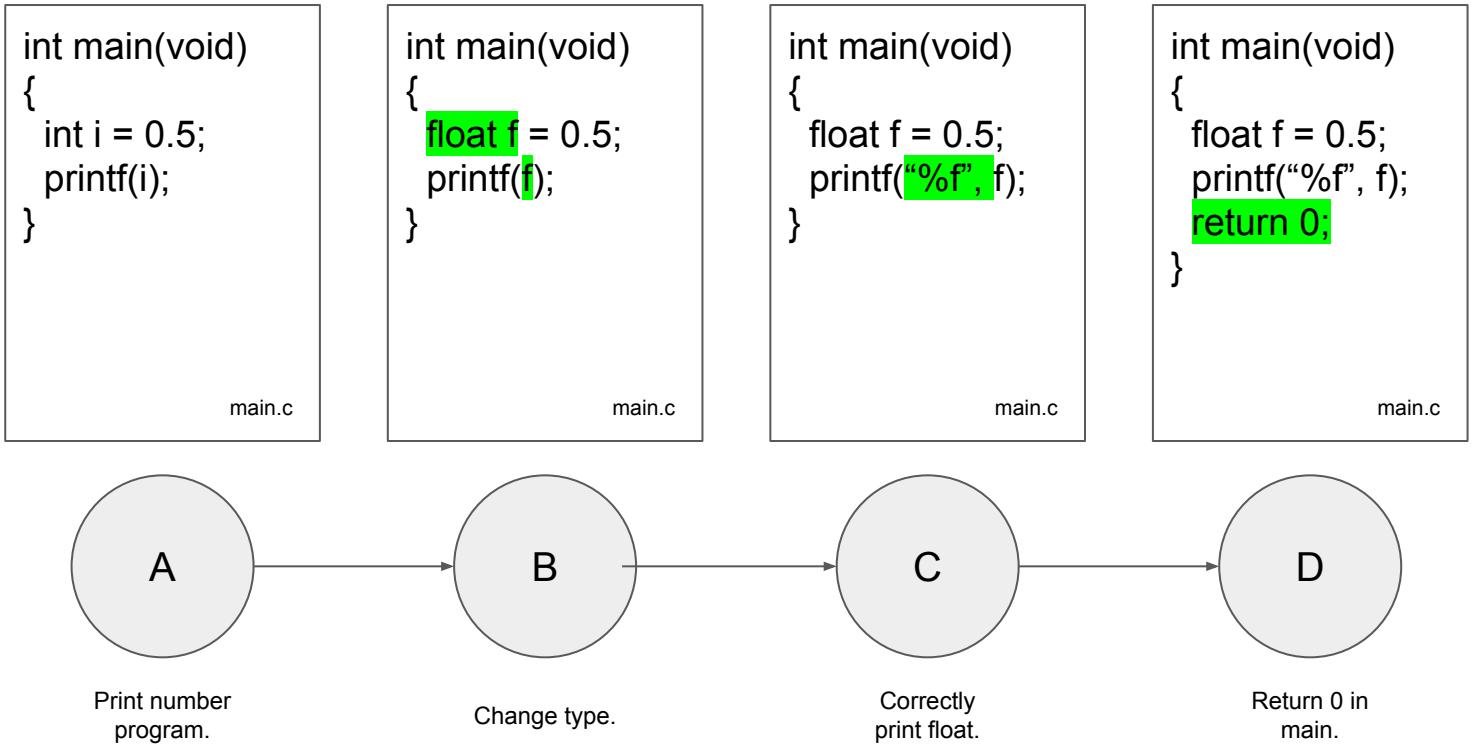
- Git
  - A distributed version control system for tracking changes in computer files.
    - We will use it to track changes in source code.
  - Useful for collaboration between developers working on the same project.
  - Useful for solo-developers who want to maintain their sanity.
- Github
  - Internet hosting service for Git repositories.
  - All assignment and lab exercises will be delivered through Github.

Terminology

- Repository: A structure that keeps track of the history of a Git project.
  - Local repository: A repository that exists in your computer.
  - Remote repository: A repository that exists on some server, in our case on Github.
- Branch: An independent line of development.
  - Every project has at least one branch, usually named *main* or *master*.
- Commit: A snapshot/milestone of a Git project.
  - Important moment in the history of development.
  - Should be based around logical units of change.

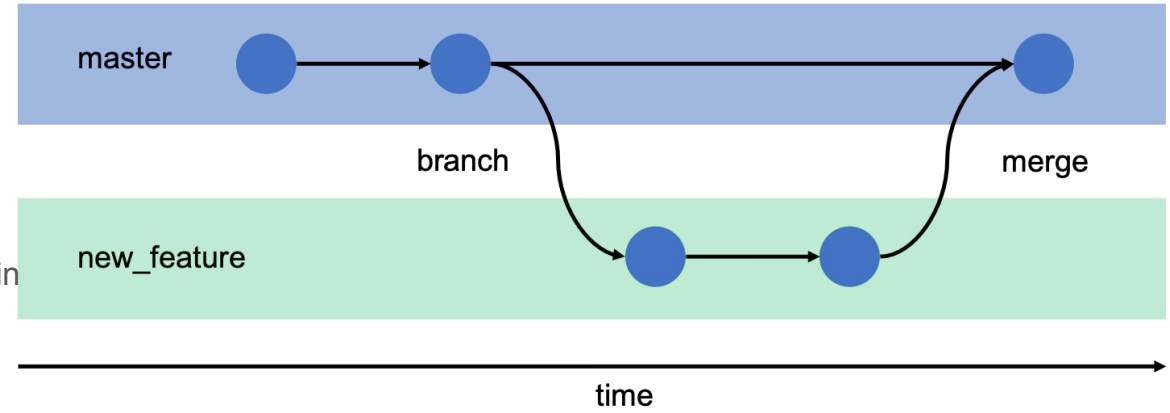
Commits

- Commits are snapshots of the project.
- At any time we can revert to a previous commit, i.e., a previous state of the project. This is useful for finding changes that introduced bugs.
- Each commit has a unique ID.



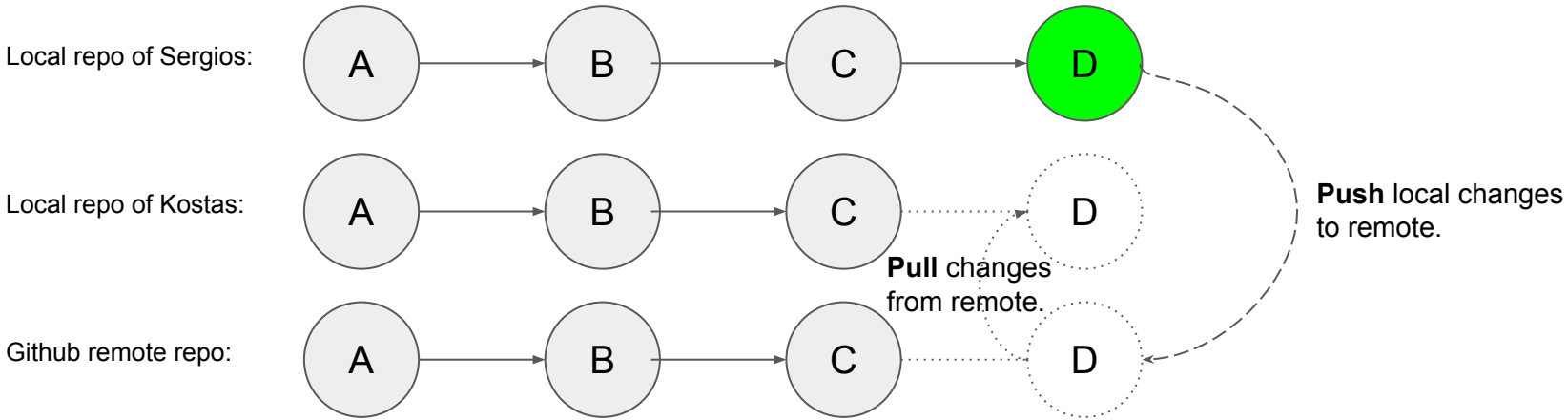
Branches

- The master branch is the “official” version of the project.
- Development, be it a bug-fix or new functionality, happens in other branches which are merged into the master branch when ready.
  - This allows for easy cooperation between people.
  - The master branch is always in a clearly defined state.
  - Merging is the process of combining two branches.
- **Using branches is not necessary for this course!**



Repositories

- A git repository (repo for brevity) holds the history of the project.
- Git is a distributed version control system, meaning that multiple repositories for the same project can exist at different locations.
  - Local and remote repositories. Usually each developer work in his local repo and **pushes** his changes to the remote repo. Other developers **pull** from the remote to get the changed code.
  - These different repositories need to be synchronized (**push** & **pull**).



Let’s git going… Prerequisites

1. Install git in your system.
   a. For Linux, install it through your package manager (e.g. `sudo apt install git`)
   b. For Window & Mac… idk :-)
2. Configure git.
   a. `git config --global user.email “your@email.com”`
   b. `git config --global user.name “Your Name”` (a decent anime btw…)
3. Create a GitHub account.
4. (?) Setup your credentials as described in:
   a. Generating a new SSH key
   b. Adding a new SSH key to your GitHub account

Creating your repos

- Create a new repo on github.com named `toast-repo`. This is your **remote**.
  - Check `Initialize this repository with a README`.
  - Its URL will be `https://github.com/<your_username>/toast-repo.
- To create a **local** copy of the **remote** repo you need to **clone** it.
  - Run `git clone git@github.com:<your_username>/toast-repo.git`.
  - A new folder named `toast-repo` containing a README file a hidden folder named `.git` will be created. The README file is the same file that was created on the remote repository and the `.git` folder is the data structure used to keep track of development history.
- Running *git remote* will show you that your local repository is connected to a single remote repository, named *origin*. *git remote show origin* will present you with more information about that specific remote.

Adding and Committing

- Modify the README.md file locally.
- Run *git status*
  - README.md appears as **modified**.
- To commit the changes run:
  - *git add README.md*
    - to add the file to the **staging** area, in other words to put it in the list of files to be committed in the next commit.
  - *git commit -m “Modify README.md”*
    - to actually commit the file.
    - the *-m* option allows you to add the commit message without needing to enter a text editor.
- Run *git status* again
- Run *git log* to see your git history

Updating your repositories, **Push** & **Pull**

- As we discussed previously, the repositories are not synchronized automatically. If you visit your remote repo you will see that the README.md file has not been updated.
- To update the remote repository run:
  - *git push*
- If the opposite situation arises, for example you update the README.md file via the GitHub interface you will need to update your local repository.
- To update the local repository run:
  - *git pull*