RMIT UNIVERSITY

Software Evolution and Maintenance

Mojtaba Shahin

Week #11: Lecture - Part 3

Topics

- **Part 3**
  - **Software Release**
  - **Version Control Systems, Git and GitHub**

Notes and Acknowledgements

- Slides/images come from the following main sources:
  - Scott Chacon, Ben Straub, Pro Git, Second Edition, Apress
  - Sebastian Rodriguez, Software Engineering Fundamentals for IT (2110), RMIT University, Course Materials on RMIT Canvas
  - Melina Vidoni, Software Engineering Fundamentals (2050), RMIT University, Course Materials on RMIT Canvas
  - Ahmed, Ashfaque, and Bhanu Prasad. Foundations of Software Engineering. 1st, CRC, 2016. Web
  - Fowler, Martin. “Continuous Integration”. Complete: https://martinfowler.com/articles/continuousIntegration.html
  - Centralized vs Distributed Version Control Systems, Mateusz Lubański,
    - https://faun.pub/centralized-vs-distributed-version-control-systems-a135091299f0

Software Release

Software Releases

- Software release is the stage when the software product has been fully developed and tested and is ready to be deployed at the customer's site.

Alpha and Beta **testing** are done here!

What do you need to do for a software release?

Deploying System

Technical Manuals

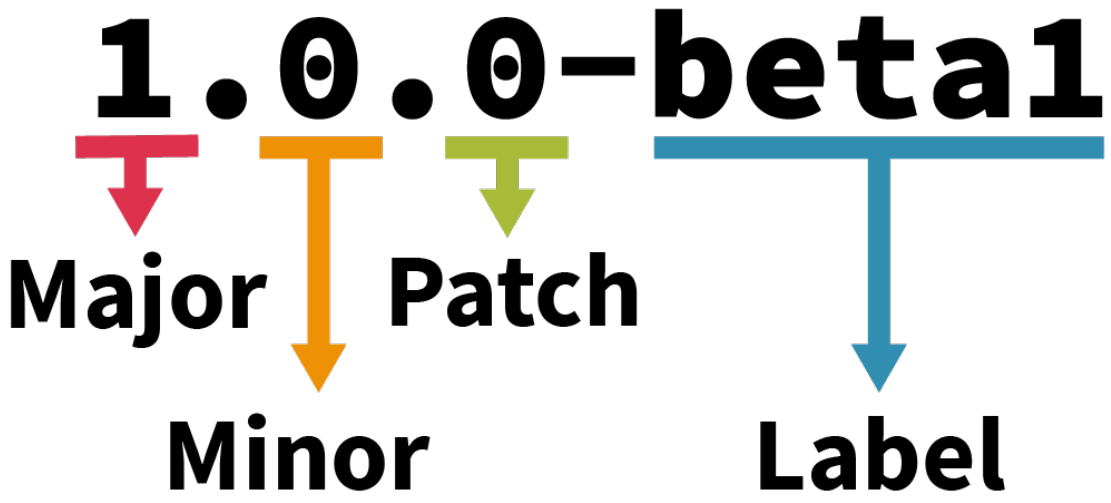
User Training

Integration with Existing Software

Other Activities

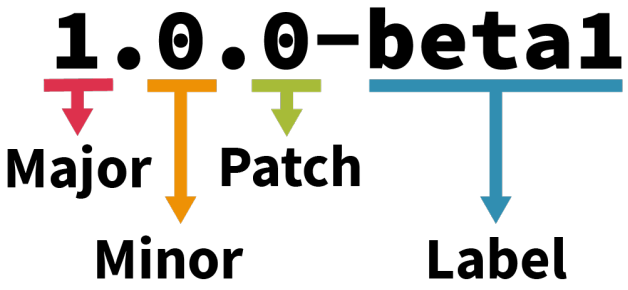
How do you identify the releases?

- **Semantic Versioning** is a standard for *numbering* the releases of a software system in a meaningful (hence, semantic) way.
- But the name is too long, so we call it **SemVer**: https://semver.org/



Semantic Versioning

- **Given a version number MAJOR.MINOR.PATCH, increment the:**
  - **MAJOR** version when you make incompatible API changes,
  - **MINOR** version when you add functionality that is compatible with the previous version (backwards-compatible).
  - **PATCH** version when you apply bug fixes that are compatible with the previous version (backwards-compatible).



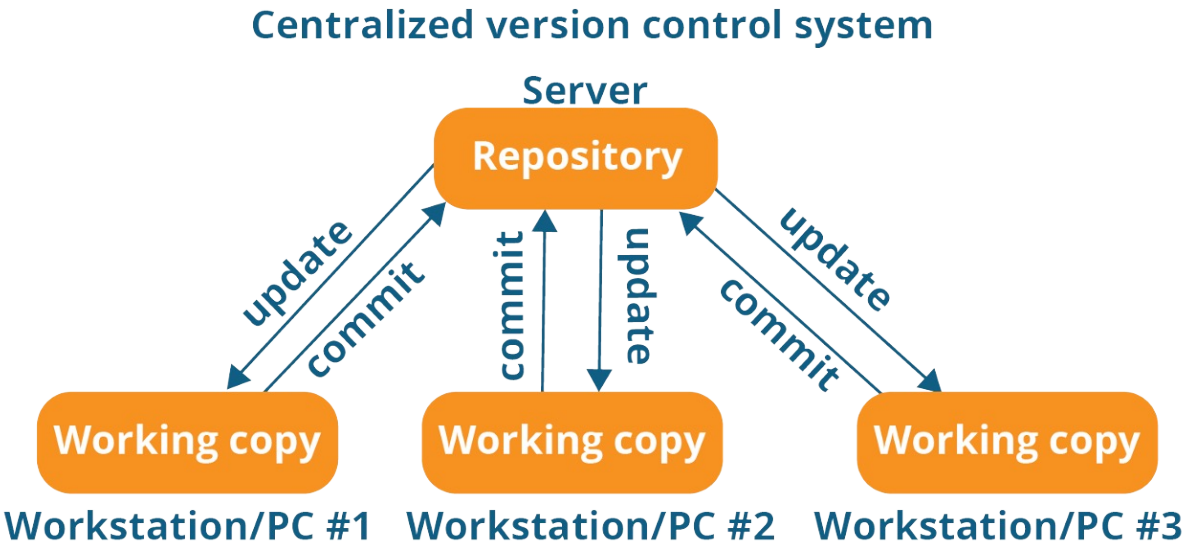
Version Control Systems, Git and GitHub

What is Version Control System?

- Version Control is a system that records changes to a file (e.g., source code, data) or set of files over time so that you can
  - revert selected files back to a previous state
  - revert the entire project back to a previous state,
  - compare changes over time
  - see who last modified something that might be causing a problem, who introduced an issue and when, and more.

Centralized Version Control Systems

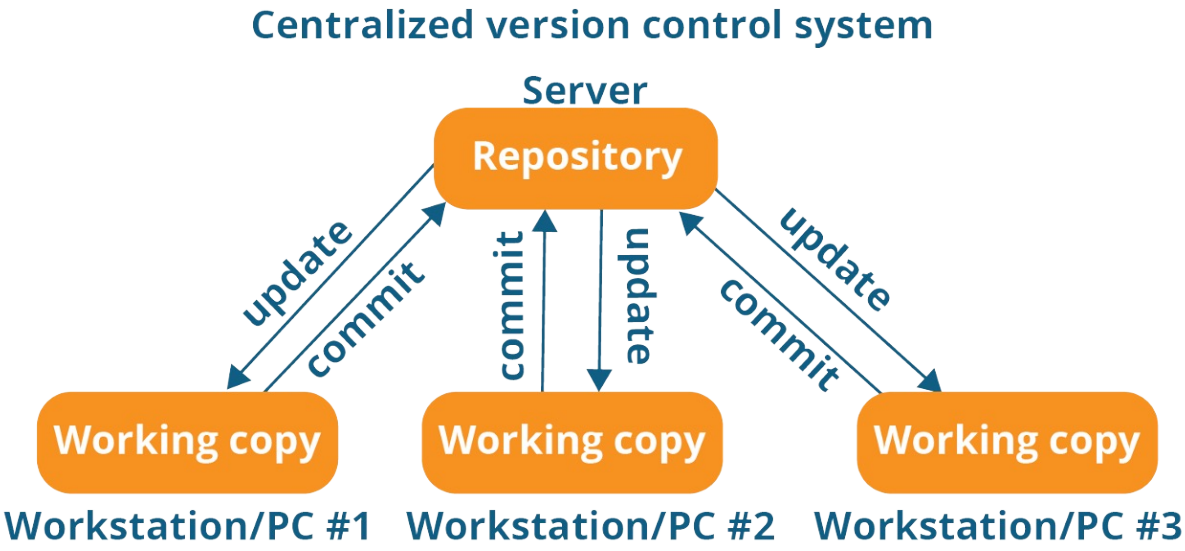
- Centralized Version Control Systems such as **Subversion** have a single server that contains all the versions of files, and a number of clients that check out files from that central place.



Source: https://faun.pub/centralized-vs-distributed-version-control-systems-a135091299f0

Centralized Version Control Systems

Programmers will "commit" their changes to this central copy.



"**Committing**" a change simply means recording the change in the central system. Other programmers can then see this change. They can also pull down the change, and the version control tool will automatically update the contents of any files that were changed.

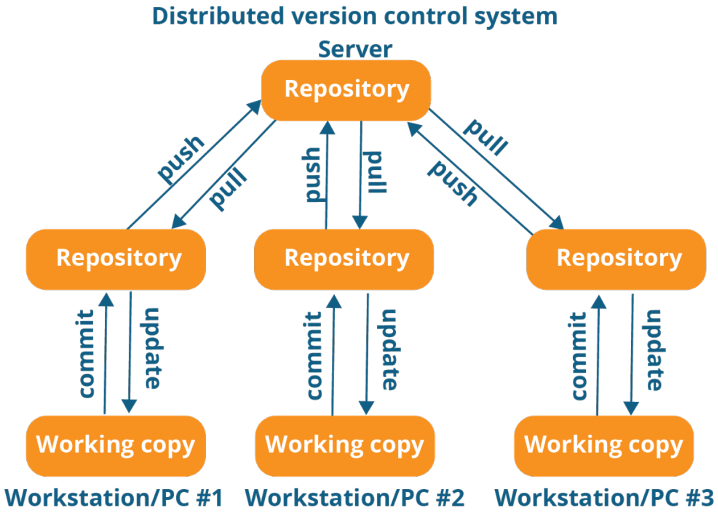
Source: https://faun.pub/centralized-vs-distributed-version-control-systems-a135091299f0

Centralized Version Control Systems

- The most obvious drawback of Centralized Version Control Systems is the single point of failure.
  - If that server goes down for an hour, then during that hour nobody can collaborate at all or save versioned changes to anything they're working on

Distributed Version Control Systems

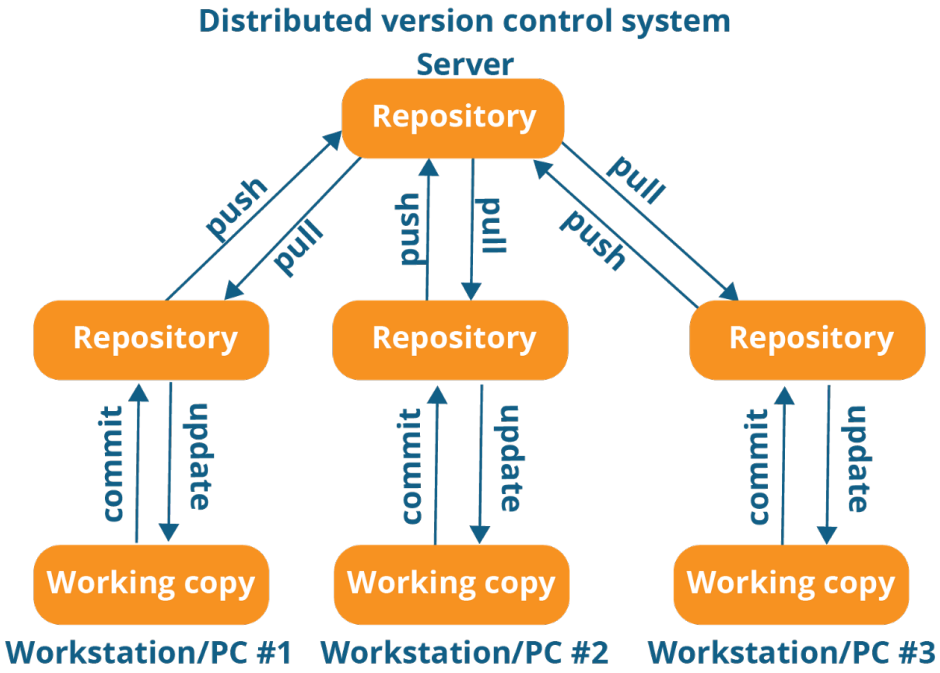
- In a Distributed Version Control System (such as **Git**), developers don't just check out the latest snapshot of the files
- Every developer "**clones**" a copy of a repository and has the full history of the project on their own hard drive. This copy (or "clone") has all of the metadata of the original.



Source: https://faun.pub/centralized-vs-distributed-version-control-systems-a135091299f0

Distributed Version Control Systems*

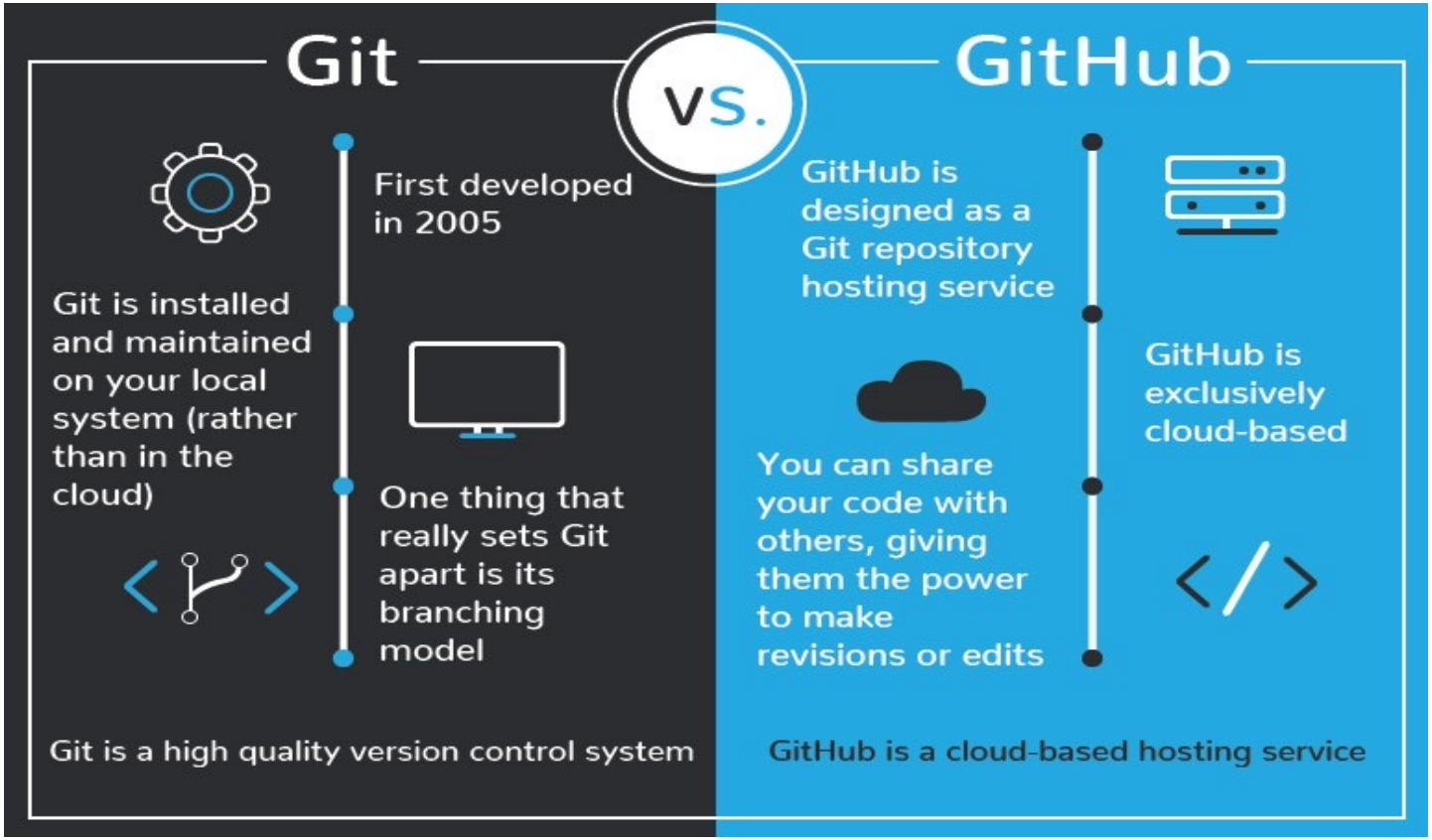
Push: While commit saves the changes in the local repository, push transfers those changes from the local repository to the remote repository such as GitHub.



Source: https://faun.pub/centralized-vs-distributed-version-control-systems-a135091299f0

GitHub

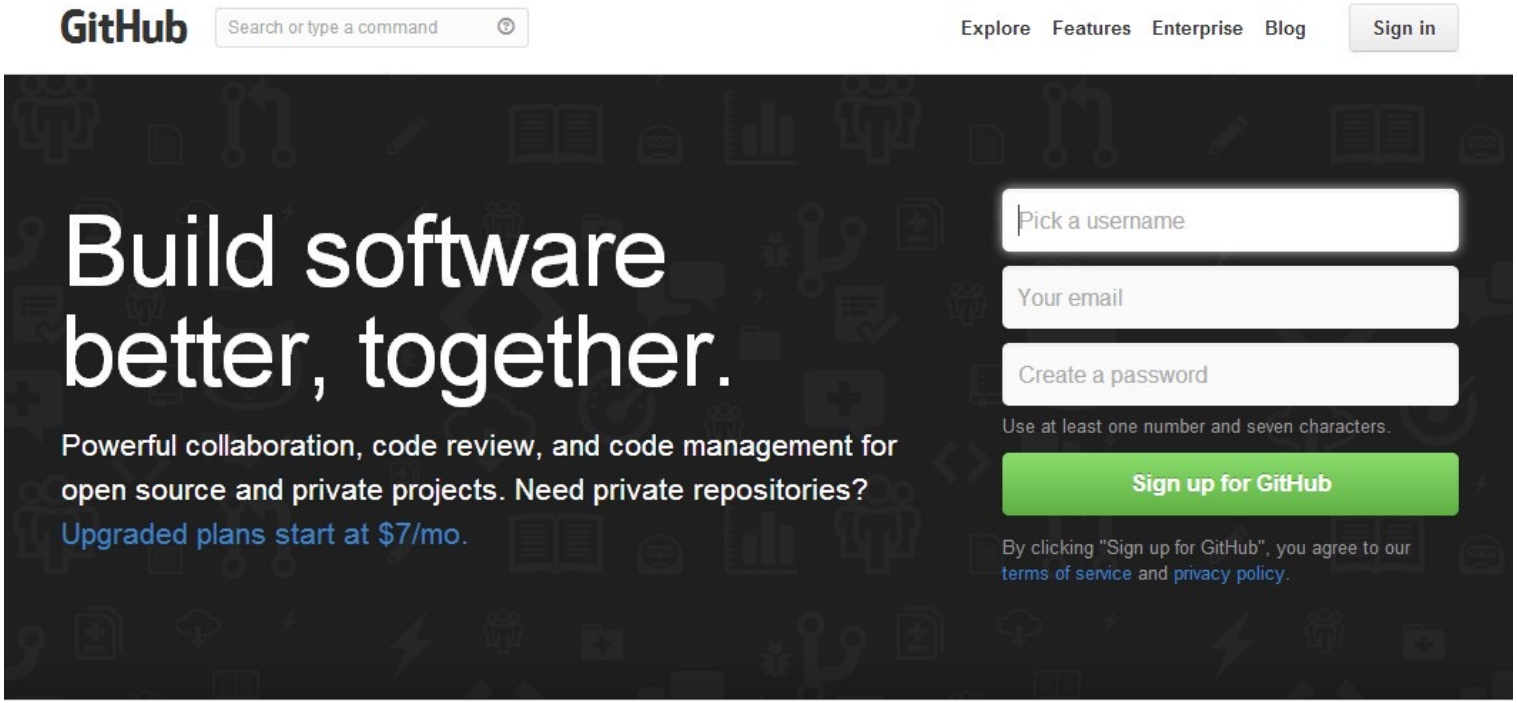
- **GitHub.com** is a cloud-based hosting service that hosts Git repositories on a remote server



Source: https://blog.devmountain.com/git-vs-github-whats-the-difference/

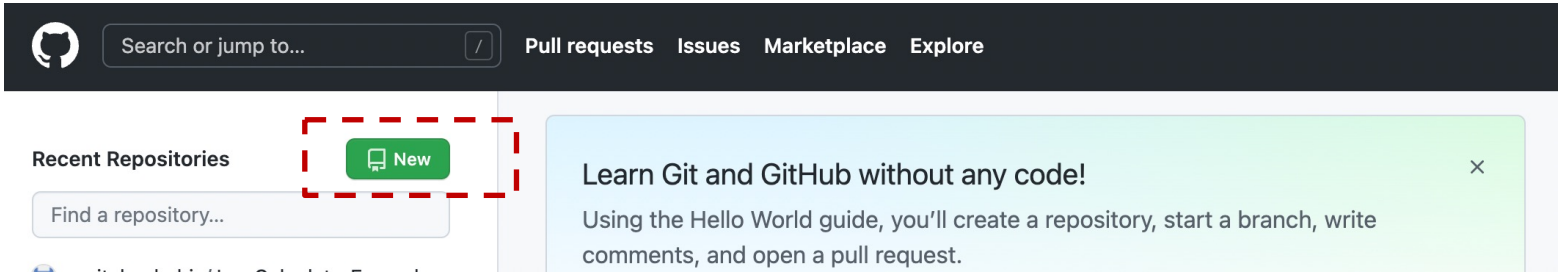
Start! Create GitHub Account

https://www.github.com



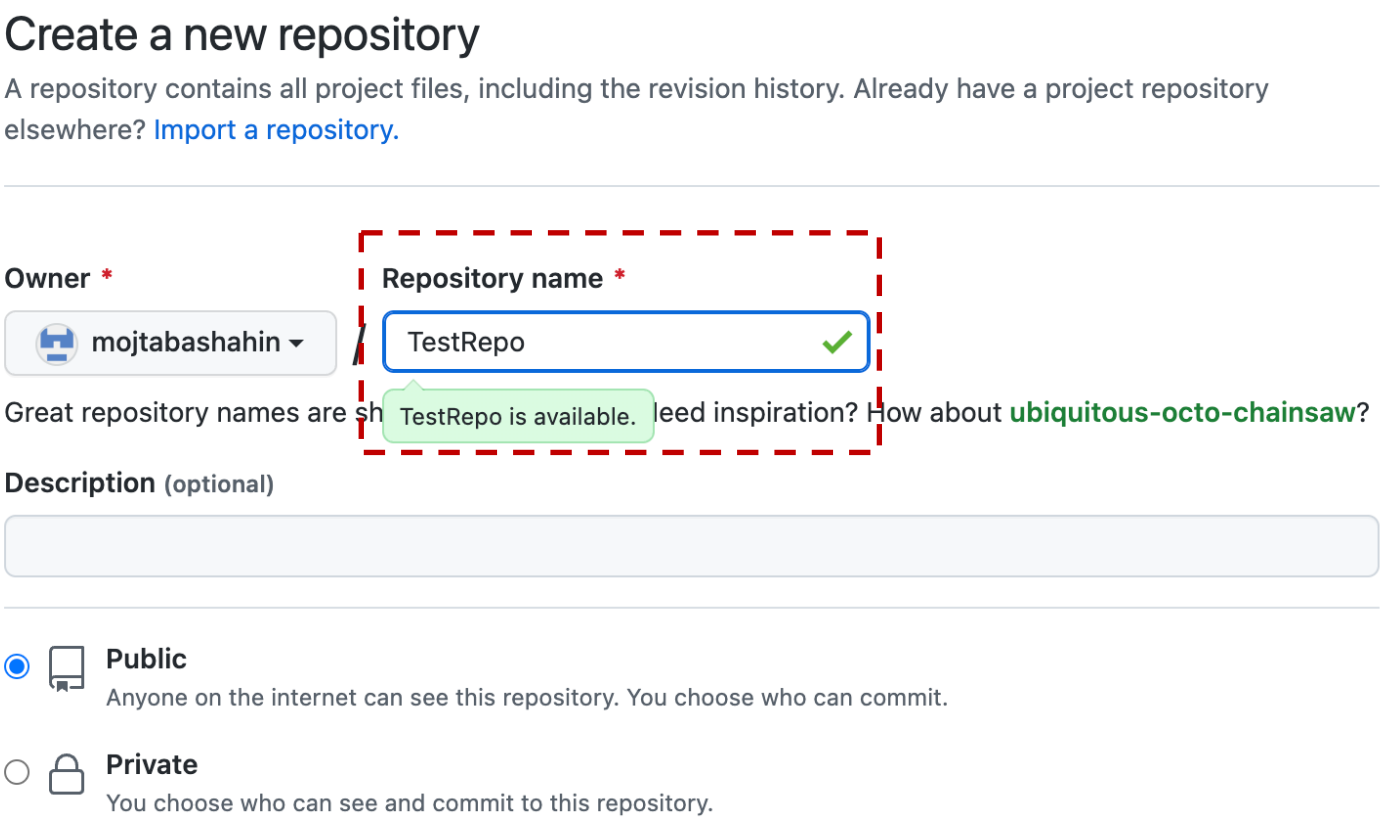
Create GitHub Repository

1) Create a new repository!



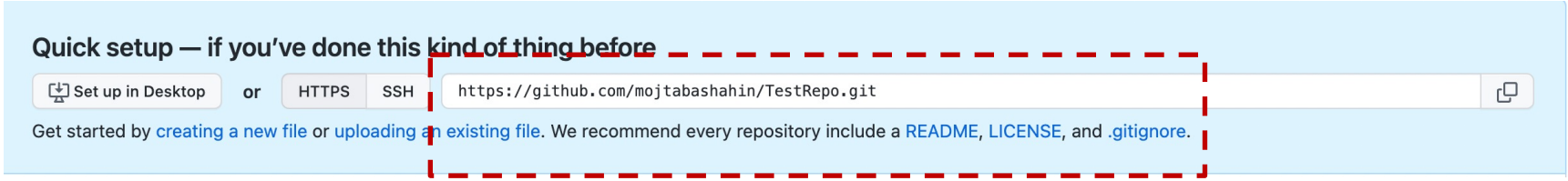
Create GitHub Repository

2) Put a name for the repository!



Create GitHub Repository

Link to the GitHub repository



Other Commands in GitHub

- ***Pull Request*** — Pull Request or simply PR is at the heart of GitHub collaboration. Supposed you made changes to your new branch. Pushed it to the remote GitHub repo so everyone else could see but no one is aware of your work as yet. Everyone is busy with their own work. **Pull Request is a way to ask other developers to review your code.**
  - A PR shows the differences between two branches in green (additions) and red (subtractions).
- ***Merge*** — Once someone has reviewed and approved a developer's changes, it's time for a merge. Merge command merges new code with the master branch.

How to Connect an Eclipse Project to a GitHub Repository

Lectorial – Week 11

References

- Scott Chacon, Ben Straub, Pro Git, Second Edition, Apress
- Sebastian Rodriguez, Software Engineering Fundamentals for IT (2110), RMIT University, Course Materials on RMIT Canvas
- Melina Vidoni, Software Engineering Fundamentals (2050), RMIT University, Course Materials on RMIT Canvas
- Ahmed, Ashfaque, and Bhanu Prasad. Foundations of Software Engineering. 1st, CRC, 2016. Web
- Centralized vs Distributed Version Control Systems, Mateusz Lubański,
  - https://faun.pub/centralized-vs-distributed-version-control-systems-a135091299f0
- Fowler, Martin. “Continuous Integration”. Complete: https://martinfowler.com/articles/continuousIntegration.html

Thanks!

Mojtaba Shahin

mojtaba.shahin@rmit.edu.au