VERSION CONTROLLING

- What and why?
- Terminology
- Best practices
- GIT

WHAT?

- Managing changes to a source.
- Changes are identified using a revision number.
- Each revision has its timestamp as well as the person who done the change.
- Revisions can be restored, compared and merged.
- “Management of multiple revisions of the same unit of information”

WHY?

- Easier backups and centralized source code repository.
- Easy collaborative development.
- Overview of changes performed to a file.
- Access control.
- Conflict resolvement.

WHAT?

- Managing changes to a source.
- Changes are identified using a revision number.
- Each revision has its timestamp as well as the person who done the change.
- Revisions can be restored, compared and merged.
- “Management of multiple revisions of the same unit of information”

TERMINOLOGY

- **Repository**
  - Central location where all the files are being kept. Usually a directory with set of files.
- **Trunk**
  - Also referred to as master branch. This is where the most stable code is being placed which is referred as the production code.
- **Stage**
  - Mark files for tracking changes.
- **Commit**
  - Create a snapshot of the changes being made to the files.

TERMINOLOGY... (CNT)

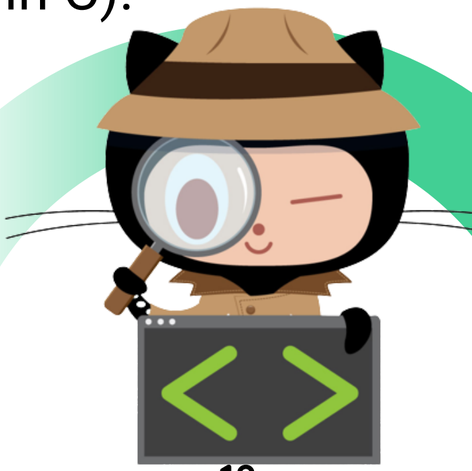
- **Branch**
  - Copy of the master branch taken at a given point. All the feature developments and bug fixes will be done in a branch. Usually it is allowed to have multiple branches at the same time.
- **Checkout**
  - Mark/unlock file for changing.
- **Merge**
  - Combining branches together to update the master branch.
- **Merge conflict**
  - Merge conflicts occur when merging a file which has been changed in two separate branches or places. Changes that interfere other changes.

BEST PRACTICES

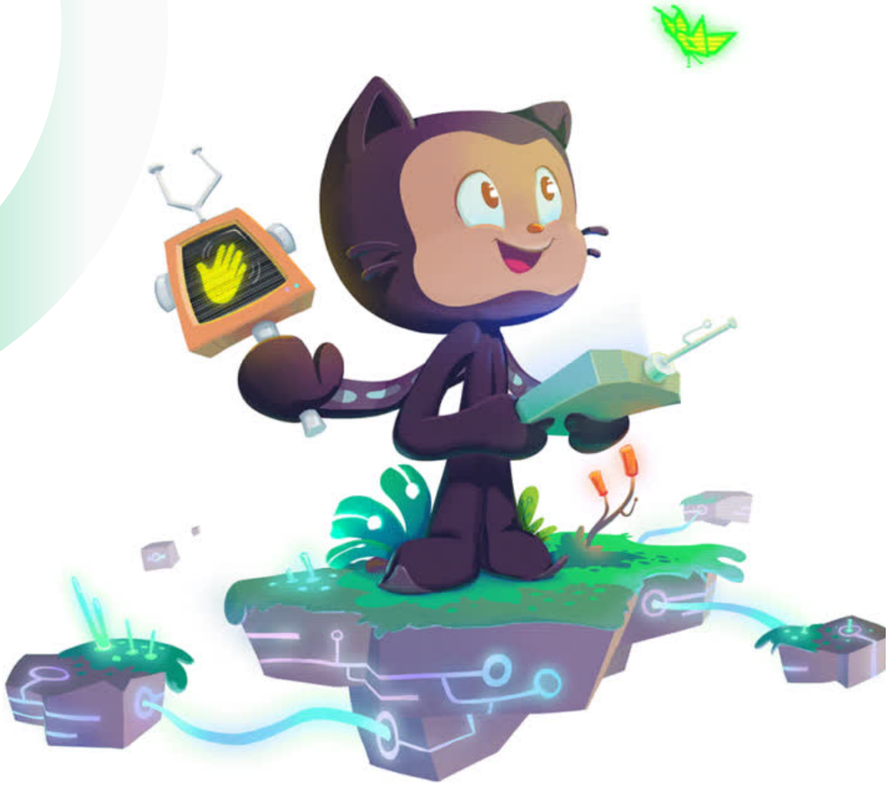
- Use a source control system.
- Always make sure to have the latest version of the file.
- In distributed source control system advice is to get the latest source code at least start of the day.
- Checkout only what you need.
- Merge code with the development branch at least once per day.
- Always make sure code is working as expected and it is not causing any other code to break.
- Follow a formal review process when merging.

GIT

- Most popular version control system.
- Distributed version control system.
  - Client get a complete clone of the source code. In a disaster situation full source along with all history can be restored from a client.
- Free and open source.
- Multiple branches and tags.
  - Feature branches, role branches (production).
- Faster comparing to other systems (works on a linux kernel and written in C).
- Support multiple protocols
  - HTTP, SSH
- Staging area, local commits and stashing.
  - Staging area - Mark files to be committed.
  - Local commit - Commit code locally without pushing into the remote branch.
  - Stashing - Keep file changes in Stash and apply them in a later.



GIT COMMANDS



- Git init
- Git clone
- Git add
- Git stage
- Git commit
- Git push

https://confluence.atlassian.com/bitbucketserver/basic-git-commands-776639767.html

GIT: INTERACTIVE LEARNING

- Following are two good interactive demos for learning git.
- The fundamentals are found in [1] and advanced branching demo is in [2].

[1] https://try.github.io

[2] http://pcottle.github.io/learnGitBranching/