Git(Social) and Doxygen

COS214

Andrew Broekman

University of Pretoria

2021

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Outcomes

After this lecture you should be able to:

- use git
- document code
- generated automated code documentation

README.md

- random three-letter combination that is pronounceable, and not actually used by any common UNIX command. The fact that it is a mispronunciation of "get" may or may not be relevant.
- stupid. contemptible and despicable. simple. Take your pick from the dictionary of slang.
- "global information tracker": you're in a good mood, and it actually works for you. Angels sing, and a light suddenly fills the room.
- "goddamn idiotic truckload of sh*t": when it breaks

What is Git?

- Source Version Control
- Distributed
- de-faco standard

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Why use Git?

- Strong support for non-linear development
- Distributed development
- Compatibility with existent systems and protocols
- Efficient handling of large projects
- Cryptographic authentication of history

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Basic Git Commands

- `clone` - bring a repository that is hosted to your local machine from a remote repo
- `add` - tracks files and changes in Git
- `commit` - save the files in Git. Add a message providing a description of the changes.
- `push` - uploads committed files to a remote repo such as GitHub
- `pull` - download changes from the remote repo and try merge into your local machine

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

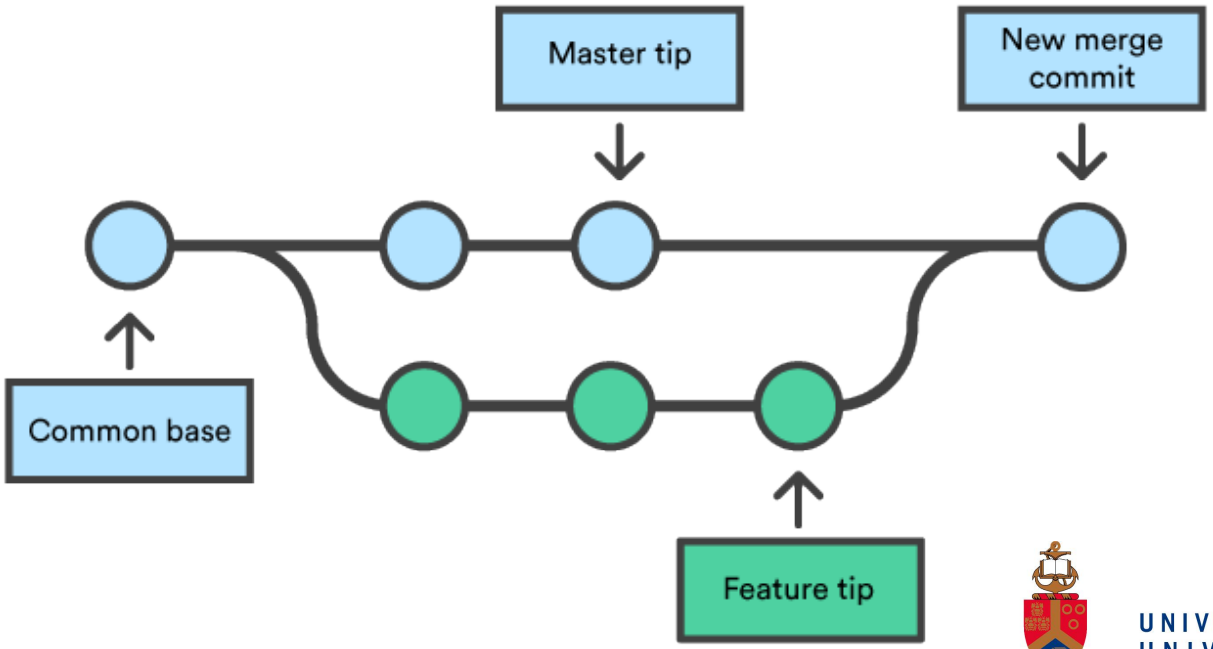
Basic Git Commands

- `init` - creates a .git directory in your current directory. This will allow you to commit files in the directory to the repo
- `status` - view the status of your files in your working directory and staging area
- `diff` - compares versions of file and highlights what changes have been made
- `reset` - use to undo local changes to the Git repo
- `help`

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

How to merge in Git?

Branch and Merge



Basic Branch Commands

- `branch` - create a new local branch
- `checkout` - switch to a local branch
- `merge` - merge the changes of the local branch to the master. Use in conjunction with `checkout`

Branch Creation

- Create a new local branch
  ```
  $ git branch name
  ```
- List all local branches
  ```
  $ git branch
  ```
- Change to a local branch
  ```
  $ git checkout branchname
  ```
- Merge changes in the local branch to the local master
  ```
  $ git checkout master
  $ git merge branchname
  ```

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

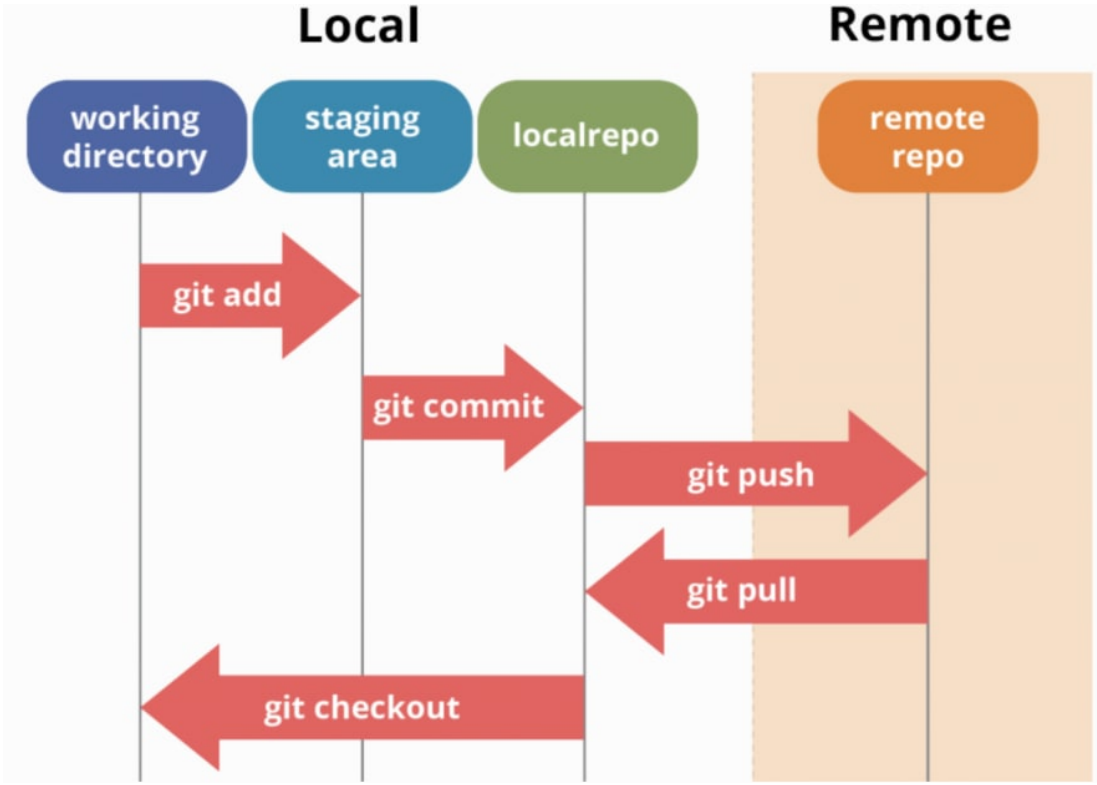
Repo Creation

- Create a new local Git repo in your local directory

```
$ git init
$ git add filename
$ git commit -m "commit message"
```

- Clone a remote repo to your current directory

```
$ git clone url localDirName
```



Source: `https://dev.to/mollynem/git-github--workflow-fundamentals-5496`

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Best Practices

- .gitignore
- `https://www.toptal.com/developers/gitignore`
- Never commit generated code (*.o, *.a, *.out, *.so)
- Large files - Git LFS (Pointers to file storage)
- Small commits
- Commit often
- CLI > GUI

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

RUN

Social Git - Git vs Git* (GitHub, GitLab, BitBucket)

- Git is the tool
- Git* is a platform where the Git repositories are hosted
- Various web platforms
- Social component - (Pull requests)
- You will need to create a new repository for your project
- You can view your change log

Hosted Git Providers

- GitHub (SaaS, 1st - Dominant)
- GitLab (FOSS IaaS/Saas, 2st - Alternative)
- BitBucket (Paid IaaS/Saas, 3st - Alternative)
- Amazon AWS CodeCommit (Free/Paid SaaS)
- Microsoft Azure DevOps (Free/Paid SaaS)

Documentation means different things to different people:

- User manual for software users
- UML diagrams for software modellers
- API documentation for coders
- ...

Types of documentation (COS301)

- Requirements - States what the client wants and what the system should do
- Architecture and Design - States how the system should be implemented
- Technical - Description of code, algorithms, libraries and APIs.
- End-user documentation - Manuals and tutorials on how to use the system

Reasons you should write docs

- Makes it easier to revisit the system after not looking at it after some time.
- Enables others to help you work on the system whether through open source or other collaborations.
- Helps third parties who might use your system to understand it
- Provides a *proof of contract* that certain aspects were implemented

We are considering *code documentation*. That is how a programmer documents their code. This includes written text and diagrams that accompanies your source code or software

- Explains how the code works
- Explains how to use the code
- Explains how to use the software

Tools for code documentation

- Doxygen
- Sphinx
- phpDocumentor
- Natural Docs
- Ghostdoc

Doxygen is a standard tool for generating documentation. It supports many programming languages:

- C++, C
- Java
- Python
- PHP and more

C++ Comments

- `// This is a single-line comment`
- ```
  /*
     This
     is a
     multi-line
     comment
  */
  ```


  ```

Doxygen Comments

- `/// This is a single-line comment`
- ```
  /**
     This
     is a
     multi-line
     comment
  */
  ```


  ```

Doxygen Tags

- Tags start with @ or \
- Depends on which version you use
- Mostly both of them work
- For example, @tag or \tag

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Doxygen Class Tags

- @file
- @class
- @author
- @version
- @brief
- @section

Doxygen Function Tags

- @param
- @return
- @todo

Doxygen goes through the comments in the code and builds documentation in various formats:

- as a series of linked HTML pages
- as Latex documentation (which can be added to the Git repo

```
#ifndef JPGREADER_H
#define JPGREADER_H

#include "imagereader.h"

class JPGReader : public ImageReader {
public:
  JPGReader();
  void access(void* buffer, double *value,
              int index, char action);
  ~JPGReader();

};
#endif
```

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

My Project

Main Page | Related Pages | Classes | Files | Search

Public Member Functions | List of all members

JPGReader Class Reference

Inheritance diagram for JPGReader:

Public Member Functions

void **access** (void *buffer, double *value, int index, char action)

Public Member Functions inherited from **ImageReader**

The documentation for this class was generated from the following files:

- jpgreader.h
- jpgreader.cpp

Generated by doxygen 1.8.16

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

- Git website - `http://git-scm.com/` Git tutorial - https://www.atlassian.com/git/tutorials
- GitHub website - `https://github.com`
- Git tutorial - `http://schacon.github.com/git/gittutorial.html`
- Youtube video on Git and GitHub - `https://youtu.be/RGOj5yH7evk`

- `http://www.doxygen.nl/`
- For more tags and their documentation check For more tags: `https://www.stack.nl/~dimitri/doxygen/manual/commands.html`

Homework

- Setup a GitHub profile and demo repository.
- For one of you CS assignments use Git either on two PCs or in two folders and push-and-pull code.
- **Challenge:** Instead of pushing code to GitHub see if you can setup a new remote as the other folder on your machine.

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Questions

Any questions with regards to:

- Source Control/ Git/ Git(Social)
- Doxygen