clearvision

Module 3

Getting Started

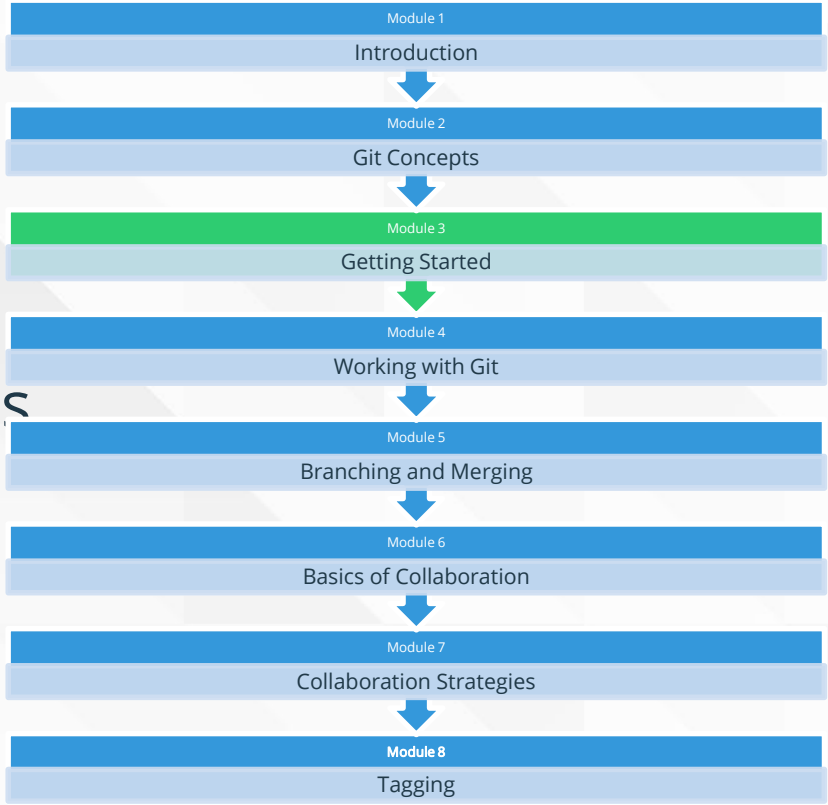
Atlassian Confluence

Module Contents

clearvision

Module Contents

- Basic Workflow
- Adding Files
- Moving or removing files
- Making Changes
- Committing Changes



git

clearvision

The Git Workflow

git

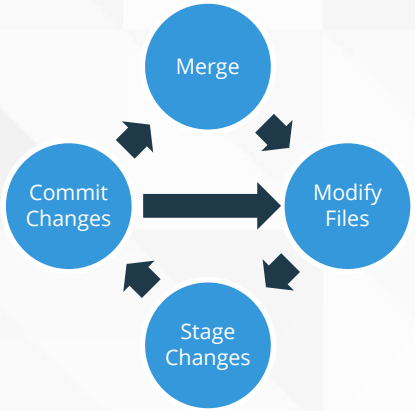
Git Workflow

clearvision

The Git workflow is slightly more complex than a centralised system.

In Git it is important to note that changes to the working directory must be staged before they can be committed to a repository.
In addition, you may need to manually merge your changes with other people

The Git workflow looks like this:
- Merge with other users (optional)
- Modify files
- Stage the changes to be committed
- Commit the changes
- Repeat



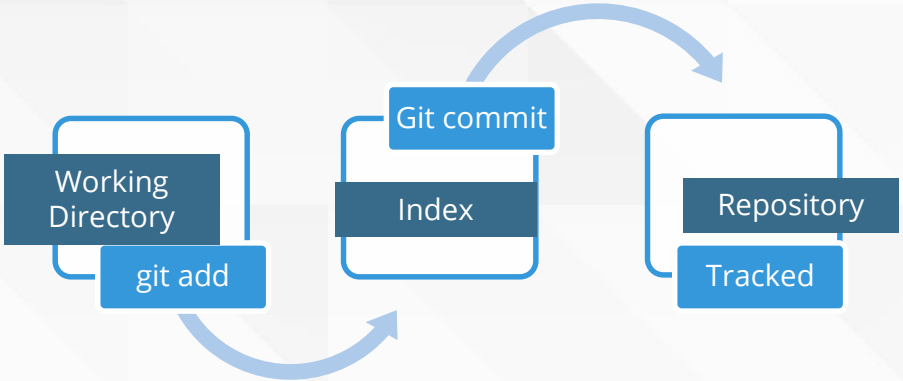
git

Adding Files

clearvision

When you have new files that you want Git to version control you can use the git add command to move them to the index

This means that these files will be included in the next commit in order to become 'tracked' by Git and monitored for changes.



git

Removing Files

clearvision

- In order to remove a file or directory you must tell Git to stop tracking it.
- You can delete a file the command *git rm*
- You can move a file with *git mv <old/file> <new/file>*
- Removing a file through any other method, such as OS commands, will not have an impact on the git repository
  - Though you can always use *git add* or *git rm* on a missing file

git

Making Changes

clearvision

Existing version-controlled files (those which have previously been “added”) can be updated in the usual manner. For example, using your preferred text editor to edit files.

Having made changes, Git will automatically flag this fact so that any changes can - if you choose - be included next time you commit your changes to the repository.

NOTE: nothing will automatically be included in the next commit - you must tell Git that you wish to include them by “staging” the change.

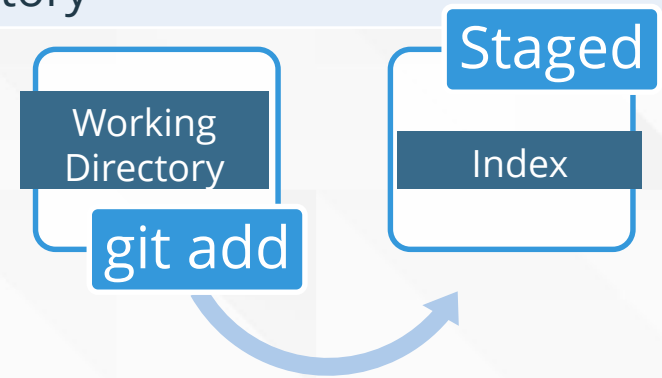
git

Making Changes

clearvision

git add has a variety of different options:

git add .	Stage all changes from the current directory (and below)
git add –I	Interactively stage changes
git add –A :/	Stage all changes including new files and deleted files
git add -u	Stage all file modifications (not new files) from the root of the repository



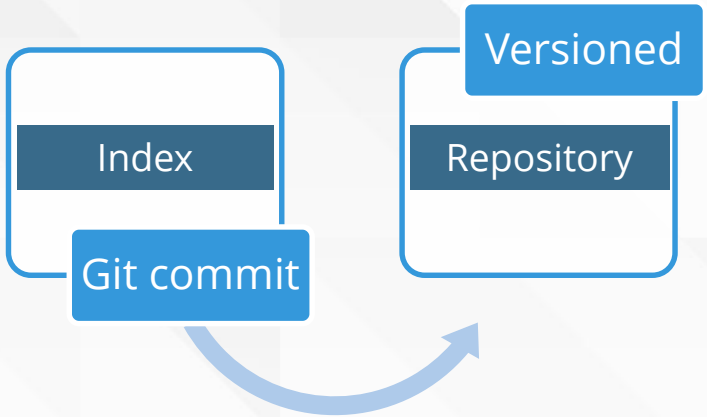
git

Committing Changes

clearvision

Once you have finished making changes and adding and removing files in the working directory you can commit your changes to the repository.

This is done using the *git commit* command.



git

Committing Changes

clearvision

git commit also has a variety of different options:

git commit	Commit all changes previously staged
git commit -m	Specify a commit message on the command line
git commit -a	Commit all previously staged changes and stage all additional file modifications (not new files)
git commit <file(s)>	Commit only the specified <file(s)>, ignoring what is in the index. This is called a “partial commit”

git

Reading History

- At any time you can see the history of commits – as they relate to your current work tree (the HEAD)
- The log command can show:
  - The history of your repository
  - The history of a specific file
  - Ancestors of specific commits

```
$ git log --format=oneline -10
a4a227a7259df19e22e45fc55b143107afadc00b Merge branch 'mi/typofixes' into maint
a5d56530e0788fa7ffcae129355ec1eb401621df Merge branch 'jh/loose-object-dirs-creation-race' into maint
4766036ecdee2d45b5a955409da5e6751c3e2f05 Merge branch 'jk/two-way-merge-corner-case-fix' into maint
66c24cd8a4b8db93b5fee81f144d6411ac7e5391 Merge branch 'sb/sha1-loose-object-info-check-existence' into maint
c8b928d7700b581118a7a1beb957184fcbe57d06 Merge branch 'nd/magic-pathspec' into maint
3e7b066e2229f45d633164beaeb8815b0973a9b4 cmd_repack(): remove redundant local variable "nr_packs"
a155a5f075cdc09e584a58d68bdce0c80e6c4b5a Git 1.8.5.1
2951add0e908f04fd63b48b8fcc138ae4dd66116 ref-iteration doc: add_submodule_odb() returns 0 for success
be38bee862d628e29043b3f680580ceb24a6ba1e Sync with 1.8.4.5
2f93541d88fadd1ff5307d81c2c8921ee3eea058 Git 1.8.4.5
```

The Git Log

git log	Shows the ancestry of the current HEAD
git log –n	Shows the last n commits of the ancestry of HEAD
git log <path>	Shows the commits in the ancestry of HEAD which modified <path> in some way
git log --since="2 weeks ago"	Show all commits from the last two weeks which are considered ancestors of HEAD
git log --branches <branch1>..<branch2>	Show all commits which are present in <branch2> but not <in branch1>
git log --not --remotes=<remote>	Show commits which are present locally but not in <remote>

clearvision

Lab Exercise

- Getting Started

git

clearvision

End of Module

git