**Branch**

* A branch is a parallel version of your Repository.
* It is contained within the repository but does not affect the master branch, allowing you to work freely without disrupting the live version
* When you have made changes to your branch, you can merge your branch back into the master branch to publish your changes
* Branch is an independent line of development

**Clone**

* A Clone is what you get, when you clone a remote repository and receive a local copy for your own modification.
* Git will keep track of all your modifications locally.
* With this clone you can edit the files in your preferred editor and use git to keep track of your changes without having to be online.
* It is however connected to the remote version, so that changes can be synced between the two when required.
* You can push your local changes to the remote to keep them in sync when you are online

**Fetch**

* Fetch is a pull without a merge.
* Fetching refers to getting the latest changes from a repository ( either local repository or online repository like github.com )
* Once these changes are fetched you can compare them to your local branches, the code residing on your local machine.

**Merge**

* Merging takes the changes from one branch and place them into another branch
* It can be in the same repository or from a different Fork.
* Merging often happens as part of a pull request which can be thought of as a request to fetch and merge

**Pull**

* A pull is a request to fetch and merge the changes.
* When you do a pull the latest changes are merged.
* for instance, if someone has edited the remote file which you are both working on you will want to put in those changes you were local copy so that it is up to date

**Push**

* Pushing refers to sending your committed changes to your master Repository.
* ( Master repository can be a local repository or a remote Repository such as github.com )
* if you want your files to be applied on the remote repository you want to do a push

**Checkout**

* Checkout lets you navigate between branches.
* The checkout command lets you navigate between the branches created by git branch
* Checking out the branch updates the files in the working directory to match the version stored in that branch and it tells git to record all new commits on that branch

**Head**

* HEAD represents your current working branch
* The head pointer can be moved to different branches
* bags all commits using get check out

**Gitignore**

* To exclude certain files or file types being tracked by git, git uses a file called .gitignore.
* The file or file types which need to be excluded are to be listed in the .gitignore file.
* GIT will ignore the files or file types listed in .gitignore from being tracked.

**Commit**

* A git object, a snapshot of your entire repository compressed into a SHA.

**Staging**

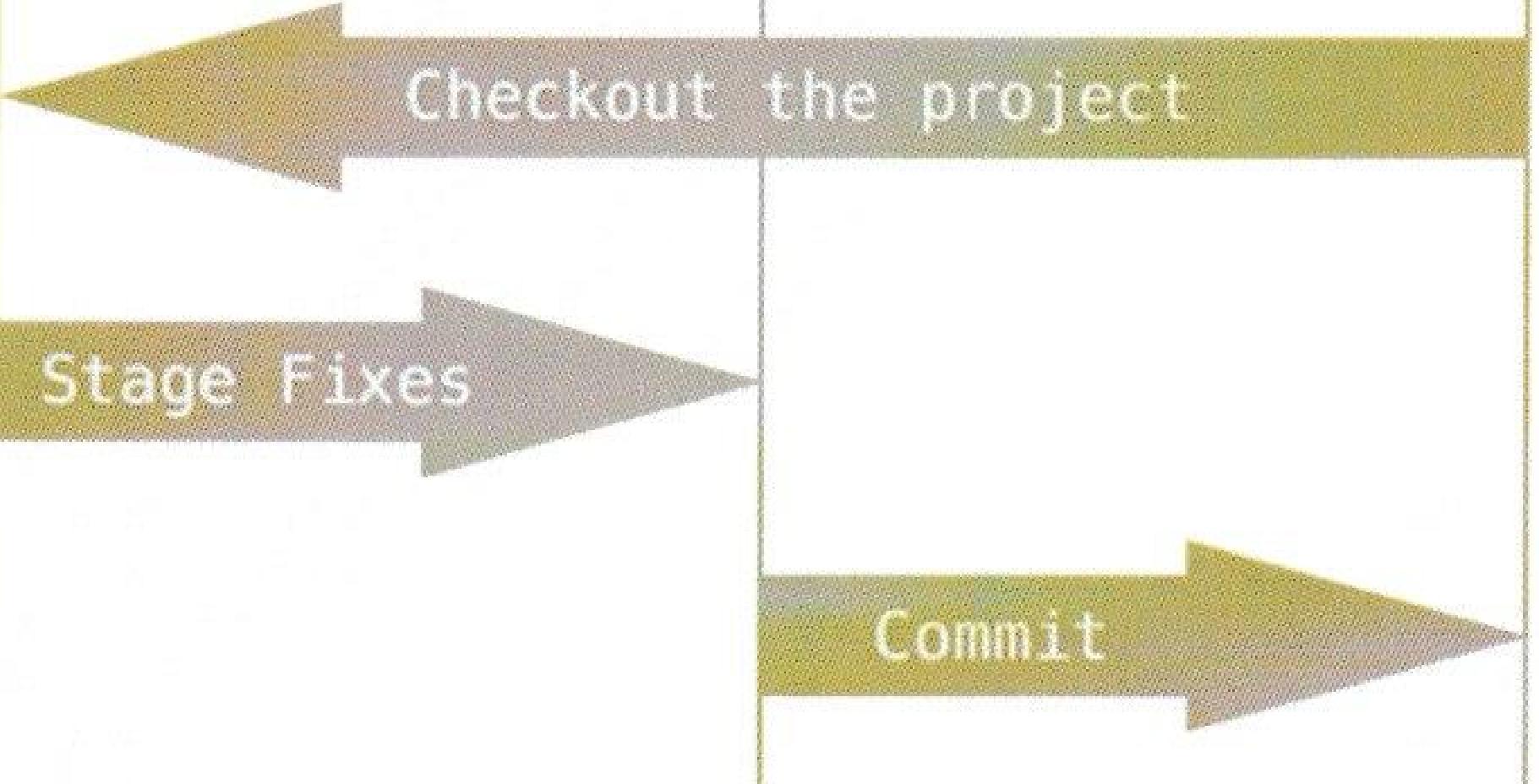
* Before committing the modified contents are moved to the staging area.

**The Three States of GT**

1. Modified
2. Staged
3. Committed
4. **Modified**
   1. Modified means that you have changed the file but have not staged it for committing it to your database yet.
5. **Staged**
   1. Staged means that you have marked a modified file in its current version to go into your next commit snapshot.
6. **Committed**
   1. Committed means that the data is safely stored in your local database.



| Working  Directory |  | Staging  Area | igit directory (Repository) |
| --- | --- | --- | --- |
|  |  |  |



**Git Directory**

* The Git directory is the directory where Git stores the metadata and object database for your project.
* This is the most important part of Git, and it is what is copied when you clone a repository from another computer.

**Types of Local Repository, Online repository**

* Local repository - for single user
* Local repository - for multi users
* Remote repository - for multi users - ( shared storage folder. )
* Online repository - for multi users with added features like Access control.
  + Github/Bitbucket - are online repositories . 5GB of space on their SAN storage. Give access to many persons, through Access control.

Development is done in different branch

Bug fixes are also done in different branches.

**What branch production ready applications run on?**

Master Branch

**Master Branch**

* Master Branch is a permanent branch, always going to reflect production ready state.
* Highly discouraged to develop on master branch

Commands

Git rm --cached index.html to unstage a staged file

Git add .

Git status

Git log

HEAD reference point to another reference

HEAD represents the current working branch.

How many branches are there in the repository?

Git branch

How many branches are there in the repository and which branch we are in?

Git branch

\* \* in the output specifies the branch you are in.

Git branch user-authentication - creates a new branch user-authentication

Git checkout user-authentication - moves to the user-authentication branch

Git diff index.html

Git checkout master

Git merge user-authentication

After merging to master branch

Git tag -a v1.0.0 -m “Version 1.0.0 release”

Git tag

GIT is DVCS

Distributed version control system

SSH keypairs are used to authenticate yourself on servers.

Git config --global --add difftool kdiff3

Git config --global --add difftool.kdiff3.path /path/to/kdiff3

Git config --global --add difftool.kdiff3.trustExitcode false