**GIT**

**GIT**

1. It is a debox tool used for source Code management.
2. It is open-source software.
3. It is used to track different versions of the code.
4. It is unique spaced / windows / MacOS placed platform.
5. No internet connection for working on individual machine.
6. GIT client communicates Git server with the help of HTTP, SSH or some protocol of GIT.

The major difference between GIT and any other VCS is the way GIT thinks about its data, other system store information as a list-based changes. These systems are CVS, Subversion, Perforce, Bazaar and so on. It like information store as a set of files and the changes made to each file over time, these types of VCS are known as delta-based VCS.

Git saves data more like a series of snapshots in a mini file system. With GIT every time you commit or saved a state of your project git basically takes a picture of what all your files look like at that moment and stores a reference of that snapshot.

To be efficient if file have not changed GIT does not store the file again. GIT stores data more like a stream of snapshots. GIT more like a mini file system with some powerful tools built on top of it rather than simply an VCS.

**FUNCTIONALITY OF GIT**

1. Every operation on Git id local.
2. Most operations in git need only local files and resources to operate. Generally, no information is needed from another computer on your network.
3. Entire history of your project right there on your local disc. For browse, the history of your project, Git does not need to go out the server to get the history and display it for you. It simply reeds it directly from your local database. Changed file also get from local database.

**GIT HAS INTEGRITY**

1. Everything in GIT is checksum before it is stored and is then refered to by that checksum. So, it is impossible to change the content of file without GIT knowing about it.
2. You can't lose information in transmition or get file corrupt without GIT being able to detect it. It uses SHA-1 hash protocol. Git generally only adds data to the GIT database.
3. In other VCS you can use or mess up changes you have not committed, but in GIT after you comment a snapshot then you didn’t lose data.
4. VCS allows you to revert selected files back to a previous state, revert the entire project to previous state, compare change overtime, see to last modified something that might be causing a problem, who introduced issue when and more.

**WORKING OF GIT**

|  |  |  |
| --- | --- | --- |
| WORKING OR WORKING TREE | STAGING AREA | GIT DIRECTORY OR REPOSITORY |
| Untracked files or modified tree. | Tracked files which need to be added in git repository. | Finally commit and stored as screenshots in git repository. |

+09

**DIFFERNET VCS**

1. **LOCAL VCS-** In local VCS a simple database that kept all the changes to files under revision control. One of the most popular VCS tools was a system called RCS (revision control system). RCs works by keeping patch sets (i.e. the difference between the files) in a special format on this hardisk. It can then recreate what ant any file look like at any point in time by adding up all the patches.
2. **CENTRALISED VCS-** In centralized VCS having a single server that contains all the versions of all the files and a number of clients that checkout files from the central place. This has advantage over local VCS. For e.g. Everyone knows to a certain degree what everyone else is doing on the project.
3. **DISTRIBUTED VCS-** In distributed VCS such as Git, Mercural, Bazaar, in this every machine have the full history if server crash then client repositories can be copied back up to the server to restore it. In distributed VCS every machine has a full backup of the full data.