VC

About Version Control What is “version control”, and why should you care? Version control is a system that records changes to a file or set of files over time so that you can recall specific versions later.

It allows you to revert selected files back to a previous state, revert the entire project back to a previous state, compare changes over time, see who last modified something that might be causing a problem, who introduced an issue and when, and more. Using a VCS also generally means that if you screw things up or lose files, you can easily recover. In addition, you get all this for very little overhead

History:

In 2005 where the community that developed the linus kernel and partner BitKEEPr broke of f- they created this sytem.

Some of the goals of the new system were as follows: • Speed • Simple design • Strong support for non-linear development (thousands of parallel branches) • Fully distributed • Able to handle large projects like the Linux kernel efficiently (speed and data size)

With Git, every time you commit, or save the state of your project, Git basically takes a picture of what all your files look like at that moment and stores are ference to that snapshot. To be efficient, if files have not changed, Git doesn’t store the file again, just a link to the previous identical file it has already stored. Git thinks about its data more like a

**stream of snapshots**.

The basic Git workflow goes something like this: 1. You modify files in your working tree. 2. You selectively stage just those changes you want to be part of your next commit, which adds only those changes to the staging area. 3. You do a commit, which takes the files as they are in the staging area and stores that snapshot permanently to your Git directory. If a particular version of a file is in the Git directory, it’s considered committed. If it has been 16 modified and was added to the staging area, it is staged. And if it was changed since it was checked out but has not been staged, it is modified

<https://git-scm.com/download/win>

$ git config --list --show-origin

$ git config --global user.name "John Doe" $ git config --global user.email [johndoe@example.com](mailto:johndoe@example.com)

$ git init

git push --set-upstream origin newBranch

show you the current repo working with

git remote -v





