INSTALLING GIT

1. Mac : http://sourceforge.net/projects/git-osx-installer/

2. Windows : http://msysgit.github.io/

3. Linux : apt-get install git-core OR yum install git-core

ABOUT GIT

1. Git is a version control tool that saves changes to groups of files so you can revert back if needed.

2. There are different types of version control tools

a. Local Version Control saves changes to files in a database

b. Centralized Version Control saves changes to a shared server

c. Distributed Version Control allows for easier sharing of files then LVC and also eliminates problems that could occur if access to the server is lost under a CVC system.

d. DVC clients have a complete backup of the files on their computer. If the server is lost the client just waits to regain contact and then uploads changes.

3. When you commit changes to files Git stores a reference of what the files look like at that moment. If a file isn't changed it isn't stored again.

4. Each client has a complete history of all changes stored locally. The client can also access all changes made to the files historically with a simple command. Also those files cannot be changed without Git knowing and changes are difficult to lose.

5. Files transition between 3 states with Git

a. Modified Files are files that have been recently changed

b. Staged Files have been marked to be saved

c. Committed Files are those that have been saved

6. Git saves all file changes to a directory as a compressed database.

a. You modify files in Working Directory

b. You notify that want to save changes in your Staging Area

c. After you Commit the file changes are saved in the Git directory

USING GIT

1. git config --global user.name "Derek Banas"

2. git config --global user.email derekbanas@verizon.net

3. git config --global core.editor "vim" # Set editor as vim

4. git config --global core.editor "edit -w" # Set editor as Text Wrangler Mac

5. git config --list # Show settings

6. git help OR git help [COMMAND] OR git help add

7. ---------------- Track a directory ----------------

a. Go to directory

b. ls -a shows all files

c. git init # Creates the .git directory

8. ---------------- Start tracking files ----------------

a. By type : git add \*.java

b. By name : git add AndroidManifest.xml

9. ---------------- Ignore Files ----------------

a. Create a .gitignore file

b. https://github.com/github/gitignore

10. ---------------- git commit -m 'Initial project version'

a. Commits the changes and sets an abbreviated commit message

11. ---------------- git status ----------------

a. Shows the state of your files meaning if they are tracked, have been modified and the branch your on.

12. ---------------- Stage A Modified File ----------------

a. Change the file and save

b. git diff # Shows what you changed, but haven't staged

c. git add AndroidManifest.xml # Stage file

d. git diff --cached # Shows what has been staged, but not committed

13. ---------------- Commit The Changes ----------------

a. commit # Opens the editor we defined above or vi

b. In vi click [ESC] i to enter insert mode

c. Type a heading that briefly explains the changes in 50 characters or less

d. Describes the original problem that is being addressed

e. Describes the specific change being made

f. Describes the result of the change

g. Describes any future improvements

h. Post a closes bug notation Closes-Bug: #1291621

i. Hit [ESC] and type wq to save and exit

j. git commit -a -m 'Changed comment' # Skips staging and commit message

14. ---------------- Remove a File ----------------

a. rm DeleteMe.txt # If you remove a file it shows as "Changed but not updated"

b. git status # If you remove a file it shows as "Changed but not updated"

c. git rm DeleteMe.txt

d. git status # Shows that the file was deleted

e. If you have committed a file to be removed you must add the -f option

f. git rm --cached DeleteMe.txt # Keep file, but remove from staging area

g. git mv DeleteMe.txt Delete.txt # Renames a file

15. ---------------- Log Commit History ----------------

a. git log # Shows all of the previous commit messages in reverse order

b. git log --pretty=oneline # Shows commits on one line

c. git log --pretty=format:"%h : %an : %ar : %s"

I. %h - Abbreviated Hash

II. %an - Authors Name

III. %ar - Date Changed

IV. %s - First Line of Comment

d. git log -p -2 # Shows the last 2 commit changes

e. git log --stat # Prints abbreviated stats

f. git log --since=1.weeks # Show only changes in the last week

g. git log --since="2014-04-12" # Show changes since this date

h. git log --author="Derek Banas" # Changes made by author

i. git log --before="2014-04-13" # Changes made before this date

16. ---------------- Undoing a Commit ----------------

a. git commit --amend # If you want to change your previous commit

b. Normally done if you forgot to stage a file, or to change the commit message

17. ---------------- Unstage a File ----------------

a. git reset HEAD AndroidManifest.xml