# 1 - Getting Setup and Started

## What you need

* <https://github.com/> Sign Up for GitHub
* Install Code editor or terminal to access Git <https://code.visualstudio.com/>
* Install Git Client on your machine
* Git Clients <https://git-scm.com/downloads/guis> if you don’t want to use command line
* Git for Windows <https://gitforwindows.org/>
* Git for MacOS <https://desktop.github.com/>
* Git download <https://www.git-scm.com/downloads> Prepare to use git on your local machine
* <https://git-scm.com/> to download latest version of git
* <https://gitforwindows.org/> for windows users get latest version
* Git for Mac as download file <https://sourceforge.net/projects/git-osx-installer/files/>

## Alternatives to GitHub for Git repository hosting

1. BitBucket
2. SourceForge
3. GitLab

## Why use git?

* source code management tool
* share collaborate and update source files
* contribute to open source projects
* track code changes and tracking
* store files and projects
* organize code
* majority of developers use git
* Work together globally from anywhere
* Project code history
* Revert back to previous versions easily.

## What does git do?

* Manage projects as repositories
* Copy and clone projects easily
* Track who made the changes
* Control Versions production and live, staging environments
* Merge versions
* Pull and push code

## Good to know about git.

* Github is a tool that uses git, they are not the same. There are other tools like bitbucket that also use git.
* GitHub is the largest host for source code anywhere
* GitHub is an excellent resource for sharing code

## Install and check

* Mac <https://git-scm.com/download/mac>
* git –version <-show git version after install on machine

# 2 - Command Line Basics

* mkdir to make a directory
* cd to change to new folder
* ls on mac or dir on windows to list folder contents
* cd .. to go up one from the current directory
* cd to go to the home directory
* rmdir to remove the directory
* touch to create a simple empty file touch code.js
* rm to remove file rm code.js
* mkdir js cp code.js ./js/code1.js
* mv to move my code.js ./js/
* clear to clear terminal
* exit to exit terminal
* tab to autocomplete

# 3 - Git Basics Create a Repo–global

* *git config --global user.name "Laurence Svekis"*
* *remove git config --global --unset-all user.name*
* *git config --global user.email "*[*gappscourses+5@gmail.com*](mailto:gappscourses+5@gmail.com)*"*
* *git config --global --unset-all user.email*
* *git init* to initialize a repo
* *git config --global --edit* to edit and see config contents
* *git status* To check current status of git repo
* *git add index.html* To add a single file
* *git add --all* To add all files or *git commit -a*
* To commit the files with a message use the -m flag *git commit -m "First Files"*
* Use the git status to check current status of the code

# 4 - Help for git GitHub repo setup

* add Add file contents to the index
* am Apply a series of patches from a mailbox
* archive Create an archive of files from a named tree
* bisect Use binary search to find the commit that introduced a bug
* branch List, create, or delete branches
* bundle Move objects and refs by archive
* checkout Switch branches or restore working tree files
* cherry-pick Apply the changes introduced by some existing commits
* citool Graphical alternative to git-commit
* clean Remove untracked files from the working tree
* clone Clone a repository into a new directory
* commit Record changes to the repository
* describe Give an object a human readable name based on an available ref
* diff Show changes between commits, commit and working tree, etc
* fetch Download objects and refs from another repository
* format-patch Prepare patches for e-mail submission
* gc Cleanup unnecessary files and optimize the local repository
* gitk The Git repository browser
* grep Print lines matching a pattern
* gui A portable graphical interface to Git
* init Create an empty Git repository or reinitialize an existing one
* log Show commit logs
* maintenance Run tasks to optimize Git repository data
* merge Join two or more development histories together
* mv Move or rename a file, a directory, or a symlink
* notes Add or inspect object notes
* pull Fetch from and integrate with another repository or a local branch
* push Update remote refs along with associated objects
* range-diff Compare two commit ranges (e.g. two versions of a branch)
* rebase Reapply commits on top of another base tip
* reset Reset current HEAD to the specified state
* restore Restore working tree files
* revert Revert some existing commits
* rm Remove files from the working tree and from the index
* shortlog Summarize 'git log' output
* show Show various types of objects
* sparse-checkout Initialize and modify the sparse-checkout
* stash Stash the changes in a dirty working directory away
* status Show the working tree status
* submodule Initialize, update or inspect submodules
* switch Switch branches
* tag Create, list, delete or verify a tag object signed with GPG
* worktree Manage multiple working trees
* *git push --set-upstream origin master* To
* *git remote add origin* [*https://github.com/svekis1/test1.git*](https://github.com/svekis1/test1.git)add the remote as origin
* *git pull origin main* Pull the files from the branch named main from GitHub

# 5 - Adding Local repo to GitHub using Git

* echo "# project3" >> README.md
* git init
* git add README.md
* git commit -m "first commit"
* git branch -M main
* git remote add origin https://github.com/svekis1/project3.git
* git push -u origin main
* TIP pull first when working locally to ensure all your files are in sync with remote GitHub file contents
* Make updates  
  1. Save File  
  2. *git add .* to add all files  
  3. *git commit -m “comment”* to commit the changes  
  4. *git push* to sync the changes to the github files

# 6 - GitHub working with git

* How to edit a file on GitHub
* Commit on GitHub
* Pull on GitHub
* Creating a branch - branching is key concept of Git
* master branch should always be deployable
* branching to make changes without affecting the main branch
* Whenever key changes are made commit the changes to the branch
* Pull request - notifies others that changes are ready. Perfect for collaboration
* GitHub Fork - a fork is a copy of the repo. Fork is an option in github to work on existing projects
* clone repo - full copy of the repo along with versions and logging

# 7 - Develop a new branch GitFlow

* Branch is a new separate version of the main repo
* With git you can duplicate the code to a new branch, make updates to the branch without impacting a live version of the code.
* Create new branches for fixes, updates and more
* Merge the new work back to the main and live branch
* You can work on different parts of the project without impact on the main branch code. Once the work is complete merge it back to the main branch
* Switch between branches to work on two versions of the project without interfering between the two
* Create a new branch *git branch new-branch*
* *git branch* to list current repo branches. \* will indicate which branch you are on
* *git checkout new-branch* to switch between branches
* to switch and create a new branch you can add the -b flag to checkout *git checkout -b new-branch1*
* *git branch -d new-branch* will delete the branch

git branch

# 8 - Merge Branches and pull Requests

* to merge switch to the master with checkout master and then *git merge new-branch*
* edi the versions and update what we want to keep
* *git add update.html*
* *git status*
* *git commit -m “merge was successful”*
* *git branch -d old-branch*

# 9 - add Commit and push and fetch

* Make the updates to the file
* Use *git add –all* to add the new files to the staging environment. This includes any new, updated or deleted changes to files.
* Once we are happy with the changes then it's time to commit the changes. Using *git commit -m “new commit”*
* *git fetch origin* Gets all the change history of the repo and updates to see the changed files
* *git merge origin/main* will combine the current branch with the specified branch

# 10 - GitPages

* <https://github.com/svekis1/mywebsite/settings/pages>
* Select the branch to deploy from
* Once set it takes up to a minute to create, and once its ready your web URL will show
* You can now make changes locally and push them to Github directly from your local computer.

# 

# 