FCPA 2022  
  
Git Every Day

Student Workbook 11A

January 25, 2022

Author

Paul Kimball  
Interface Associates

1. Git Every Day

# Git Projects

## You'll have a different Git *project* for each body of code that you work on

* + May contain many .c and .h files
  + May contain Visual Studio project files

## Your project may be tracked by more than one repository

* + **The original corporate or public repository that you cloned or copied**

#### Where the code is maintained

#### On corporate machine

#### Managed by someone else

* + **Your local repository**

#### Where you work

#### On your machine, in your file system

#### Managed with git command

* + **Your remote repository**

#### Where you share

#### On GitHub or other hosting service

#### Managed with GitHub account

#### push/pull with git command

* + **Other people's repositories**

#### When they are working on the same project

* Each repo contains shared history at the time it was copied, but can develop along separate paths over time as files are edited
  + Sometimes this is exactly what you want - this is what branches are for

Synchronizing Local/Remote

* Sometimes, you'll want to keep your local and remote repo in sync
  + Especially if you think of your GitHub repo as a "backup"

## A remote project is linked to the local project as its "origin"

* + This doesn't really say which one came first

The Git Commandments (local version)

* git config
* git init
* git status
* git branch
* git add
* git commit
* git restore
* git rm
* git mv

The Git Commandments (remote version)

* git clone
* git remote
* git fetch
* git pull
* git push

Creating a Project

## Remote-first approach

* + Create project on GitHub

Add README.md and .gitignore files

* + Commit files via web interface
  + Clone to local repo - remote "origin" is set automatically

## Local-first approach

### Create project on local machine

Create local directory

Initialize as git project

Create files or copy them into project directory

Add files to staging area

Commit changes to local repo

### Create EMPTY project on GitHub

Add no files!

### Set remote origin and branch

### Push to GitHub

## Every day

* + Work a little
  + Save your files
  + Stage your changes
  + Commit your changes
  + Synchronize local and remote repositories

git pull from remote to local

git fetch from remote to local

# Git is (clearly) Not Magic

## Git does one job - it tracks changes to your files

### Git doesn't know whether you're writing programs or fanfiction

### Git doesn't know about your project "structure" or "type"

## You have to tell git what to do

### Git doesn't create source files or VS projects for you

### Git doesn't "automatically" save your changes

## You can add git to an existing project

### But it won't know file history from before git

## You can remove git from a project -just delete the .git subdirectory

### But now it's gone, and any history with it

## You can delete your project directory and all your files

### And they're gone. Sorry.

#### You must have backups!!

## You have to tell git which files to manage

### Git will happily manage files that you don't want it to

#### You must have a .gitignore file (!)

### Git will happily ignore files you thought were important

#### You must remember to add AND commit files

## Git works best with human-readable text files

### Easier to compress and compare

### There are better tools to manage music, video, photos, etc.

## You must use git continuously, conscientiously

### Don't just leave and go home, and hope that git remembers where you were and what you were doing

### If you lie to git, it will believe you