1. Ask intro questions
   1. Previous version control experience
   2. Windows vs Mac users
   3. Powershell Users
2. Where to find tools
   1. Git
      1. https://git-scm.com/download/win
      2. http://git-scm.com/download/mac
      3. choco inst -y git
   2. PoshGit
      1. https://github.com/dahlbyk/posh-git
      2. cinst -y poshgit (run after git installed and PS restarted)
   3. Git Aliases
      1. cinst –y TimRayburn.GitAliases
      2. <https://github.com/joshrizzo/gitnaughty>
   4. MergeTools
      1. P4Merge: <https://www.perforce.com/product/components/perforce-visual-merge-and-diff-tools>
         1. Config: <https://danlimerick.wordpress.com/2011/06/19/git-for-window-tip-use-p4merge-as-mergetool/>
      2. Meld: <http://meldmerge.org/>
   5. Gitignore generator: <https://www.gitignore.io/>
3. Understanding the different areas
   1. Working
   2. Staging
   3. Repository
   4. Remote Repository
4. Starting from scratch? (No, you aren’t)
   1. Git init
   2. Generate .gitignore based on project type
      1. Watch out for Unicode!
   3. Create repository (github/bitbucket)
   4. Git remote add origin PATH
   5. Code
   6. Add
   7. Commit
   8. Push
      1. Git push –u origin master
   9. You become the team lead
      1. Decide workflow
5. Joining an existing project
   1. Get set up
      1. Git clone PATH
      2. TALK TO TEAM LEAD!
         1. Workflow
            1. Feature Branch
            2. Gitflow
         2. Pull Requests?
         3. Continuous integration?
            1. Auto-publish branches?
   2. Follow workflow
      1. Create appropriate branches
      2. Code
      3. Rebase from master
      4. RUN TESTS
      5. Merge to master
      6. Push master
      7. Clean up dead branches
         1. Git branch –merged
         2. Git branch –d DEADBRANCH
6. General use of git
   1. Adding files
   2. Committing code
   3. Rebasing from Master
      1. Git Stash
   4. Undoing mistakes
      1. Git checkout FILE
      2. Git reset head FILE
      3. Git reset head
      4. Git reset head --hard
   5. Merge conflicts
      1. Creating merge branches
7. Edge-cases of Git
   1. Cherry picking commits
   2. Rewriting history
      1. Local
         1. Forgot a file? Git commit –amend
         2. Squash: Git rebase –I HEAD~#
         3. Squash: Git reset HEAD~# && git commit –am “message”
      2. Server
         1. Git push –force
         2. Apologize to fellow developers
         3. Git pull --force