Intro to Git!

1. What is revision control?
   1. Why do I need it?
   2. Benefits for a team
   3. Benefits for a solo dev.
   4. What happens if I don’t have it?
2. Description of client/server model.
   1. Subversion, Perforce, Surround SCM, Team Foundation Server
   2. Server is the master, and all the clients are beholden to that.
   3. Client only has copies of files they pulled or checked out.
   4. Files cannot be worked on simultaneously.
3. Description of distributed model.
   1. Git, Mercurial.
   2. Everyone has a copy!
   3. Everyone is the master….in their own little realm.
   4. Simultaneous work, but now you have to reconcile.
4. How does Git work?
   1. Committing
   2. Branching/Merging (unbranching)
   3. Push (and Pull)
      1. Within a team and to non-developers (“the public”)
   4. Everything else serves these three functions.
5. Standard Operating Procedures
   1. What is GitFlow?
   2. Alternatives.
6. Servers
   1. Purpose
   2. GitHub, BitBucket, GitLab.
   3. Place in the workflow.
7. Flavors of Git.
   1. Command Line
   2. GUI: Git GUI, SourceTree, Kraken, TortoiseGit, GitHub Desktop, etc.
   3. GUI Integrations: VSCode, Atom,
8. Git with Labview.
   1. File Types
      1. XML based files
      2. .vis (binary blobs or are they?)
   2. Compare/Diff
      1. Option 1: do it manually
      2. Option 2: lvcompare.exe
   3. Merge
      1. Option 1: Override
      2. Option 2: Manually.
      3. Option 3: lvmerge.exe
   4. Ignore
      1. Files to ignore.