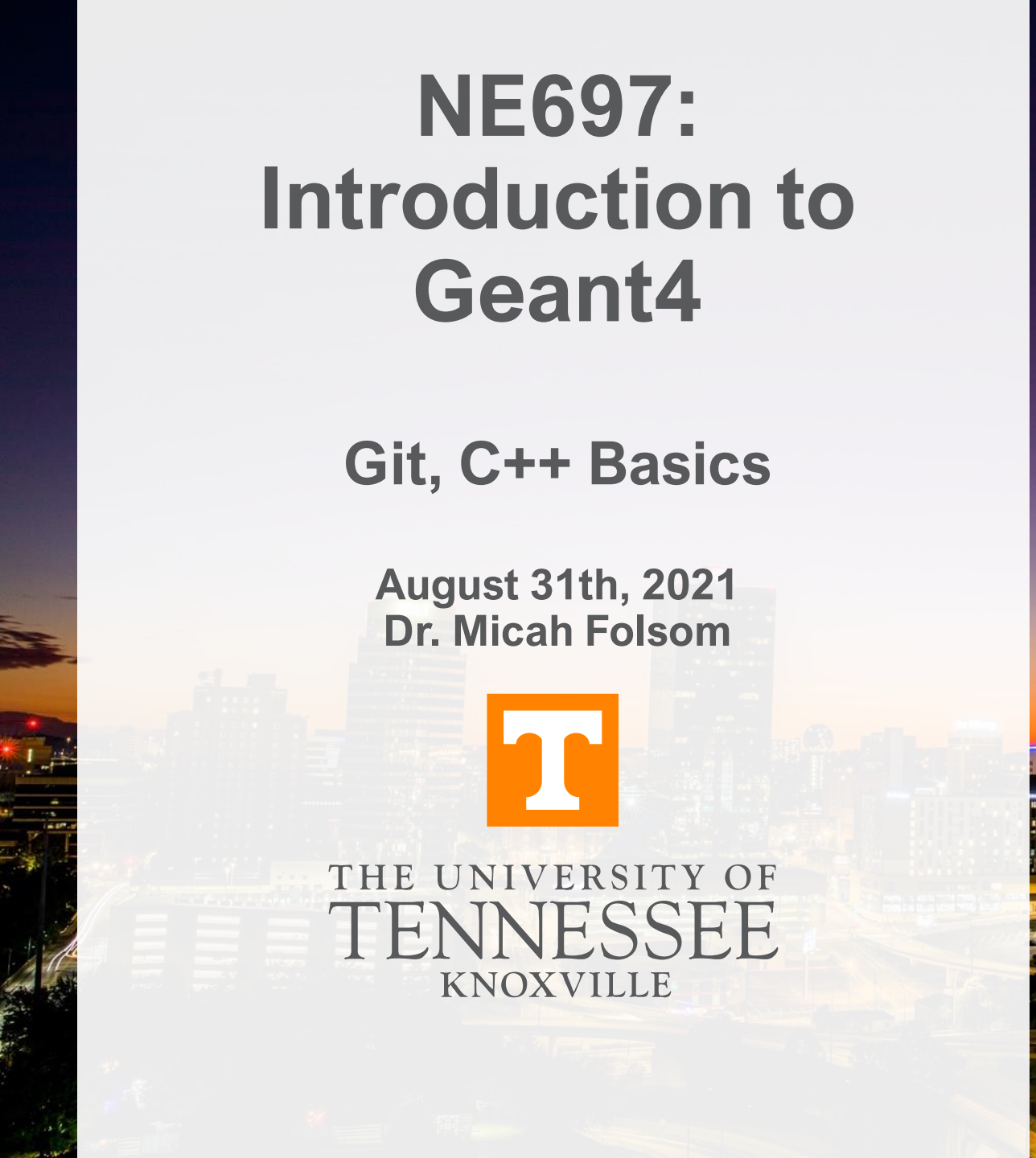
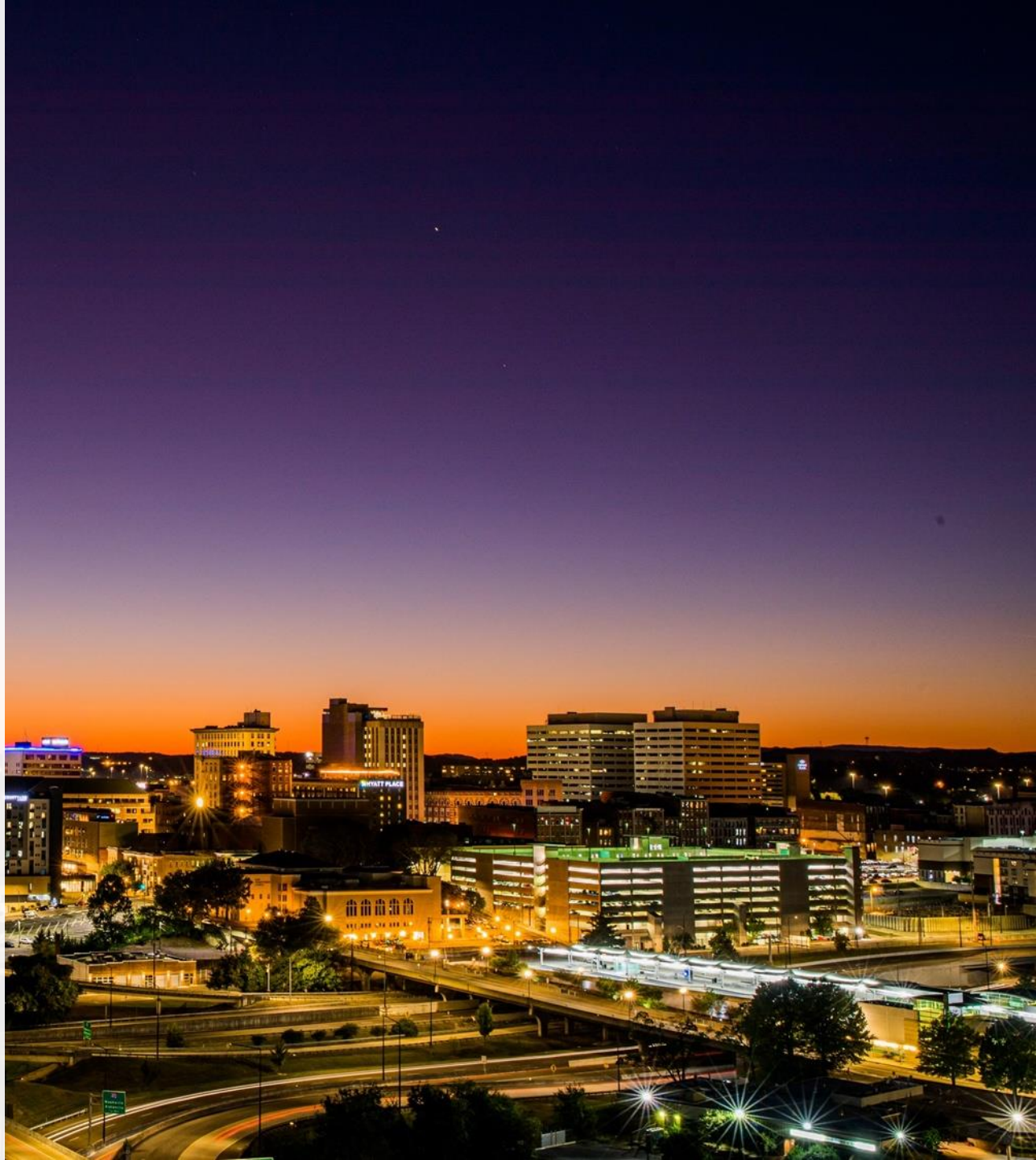
NE697: Introduction to Geant4

Git, C++ Basics

August 31th, 2021

Dr. Micah Folsom





Today's Agenda

- Server access & Github – due today!
- All set on code editors? VSCode?
- Server access for auditors
- Any other administrative items?

Reminders

- All we're doing is manipulating text files
  - g++/clang: converts text file(s) into machine code
  - CMake: converts text file(s) into build config/instructions (also text files)
  - git: a nice way to track changes to text files
  - VSCode: a fancy text file editor
- If you can dodge a wrench, …
  - If you can edit a text file, …
- C/C++: verbosity in exchange for control, safety, and performance
  - MATLAB, python, many major, performant python packages

CMake

- The hierarchy of building C++ code:
  - **g++/clang**: program that compiles the code into an executable
  - **Makefile**: scripting language for generating g++/clang commands
    - You won't need to touch these at all thanks to CMake! Yay!
  - **CMake**: higher-level scripting/configuration that generates **Makefile**s
- Includes tools for assembling lists of file paths
- Can set up build options, compiler flags, etc
- Copy files to build or install directory
  - We won't generally be "installing" since we're developing for ourselves

Git

- Version control: the (more) sane way to collaborate with code
- Free, works on every platform, it may save your life
  - Use it for your own code, even if it's just you
  - It's in ~the cloud~
  - Use it for your dissertation!
- The commands can be clunky; use the VSCode GUI
- I'll show you on the command line first, because understanding the commands is still important
- Bonus: https://www.atlassian.com/git/tutorials/what-is-version-control

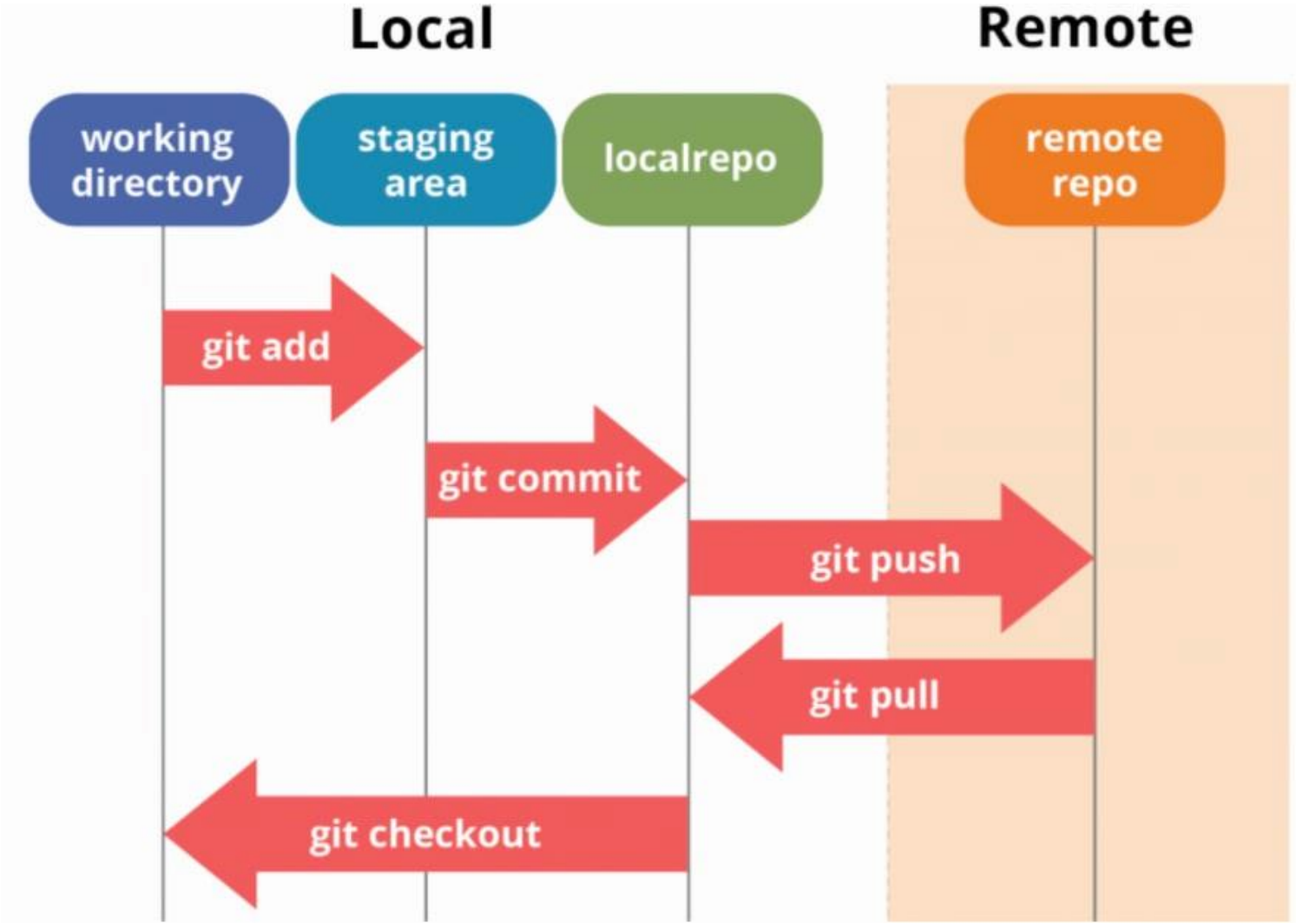
Git: Basic Terms

- **Repository**: root directory of the project, contains .git directory
- **Branch**: named history/modification trajectory of code
  - Implementing a new feature: create branch, make changes, test, merge **from** main (to get others' changes), merge **to** main
  - It's just a separate workspace within the same repository
- **Commit**: a record of a change/set of changes to the code
  - Uniquely hashed
  - Comes with a comment: "fixed that pesky timestamp bug in the DAQ"
  - Can be tagged (e.g. v1.3.0)

Git: Basic Terms

- **Clone**: get a repository and **checkout** the default branch (trunk)
- **Fetch**: get the **remote** changes in an existing repository
- **Pull**: get the **remote** changes in an existing repository and **apply locally**
- **Checkout**: change local repository state (branch, commit, or file)
- **Push**: push local changes to the **remote** repository
- **Merge**: combine 2 branches, automatically reconciling the differences where possible
  - **Merge conflict**: it couldn't be done automatically, must be done manually
- **Tag**: attach a human-readable label to a commit

Git: Basics



https://dev.to/mollynem/git-github--workflow-fundamentals-5496

Git: Basic Commands

- git clone [https:// **or** git@]
  - Download the repository and checkout the default branch (main/master)
- git checkout [commit hash]
- git checkout –b [branch] (make a new one)
- git checkout [branch] (existing)
- **git status**
  - What state am I in?
- git pull
- git commit –m “added some new code”
- git push

Git: Basic Commands

- [DEMO]
  - Make a new repo on Github, add .gitignore for build dir
  - Clone, or push local
  - Add hello.cpp
  - Branch to add CMake
  - Modifications to main
  - Merge it all together
- Command line and VSCode? Just 1?
- Adding an ssh key – needed to push!

Assignment 2

- Demonstrate the C++ development workflow we've learned using our new tools
  - Make a new folder in your Github repo called **assignment2**
  - Create a CMakeLists.txt file
  - Write a simple program (1 source file) that takes the first command-line argument *N* and prints 2^N to standard out
    - Headers for: power function and converting text to numbers
  - Commit your code and push to Github
    - Add your build directory to .gitignore if the folder is in the repo