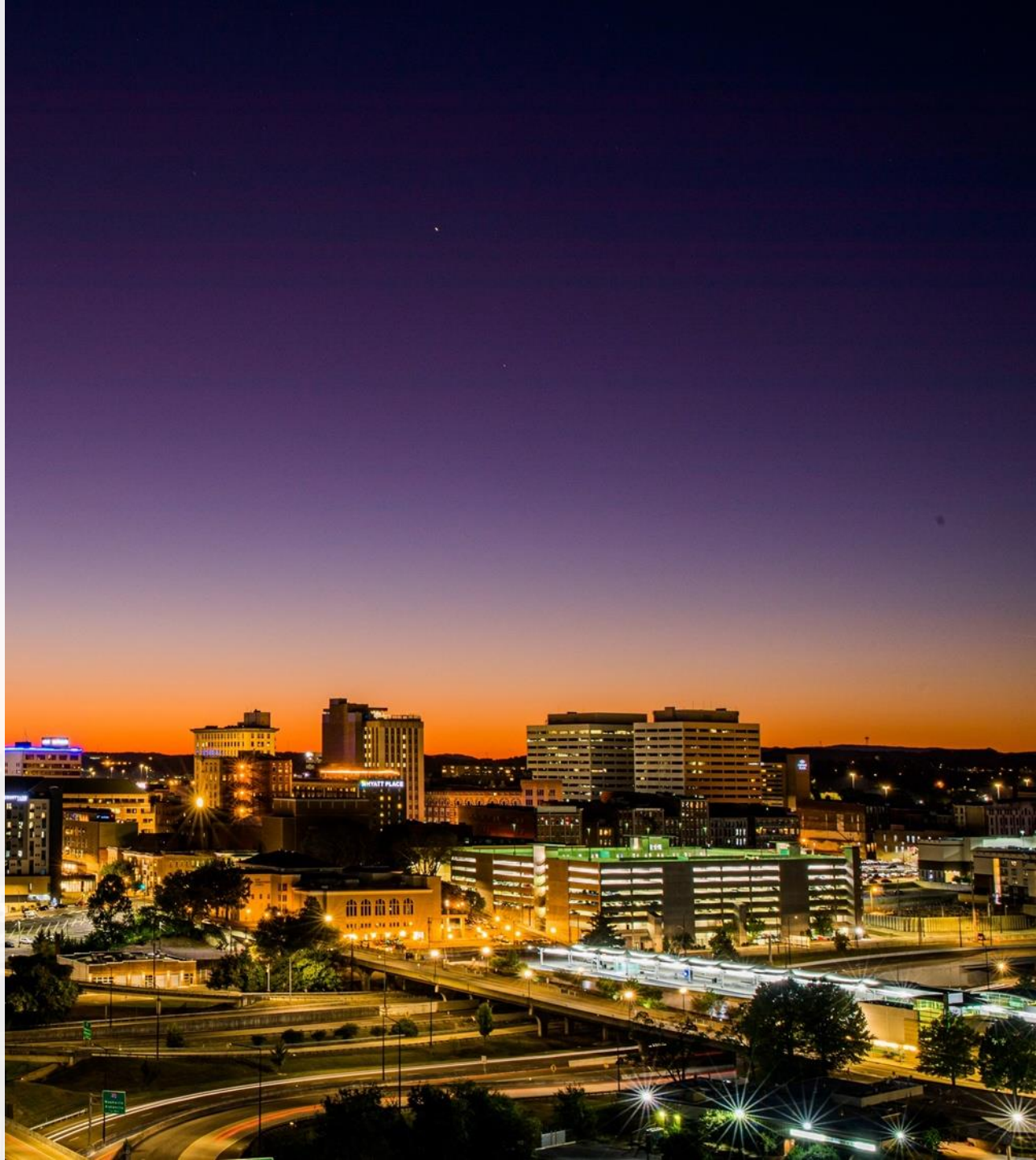
NE697: Introduction to Geant4

Geant4: UI Commands

November 16th, 2021

Dr. Micah Folsom



THE UNIVERSITY OF TENNESSEE KNOXVILLE

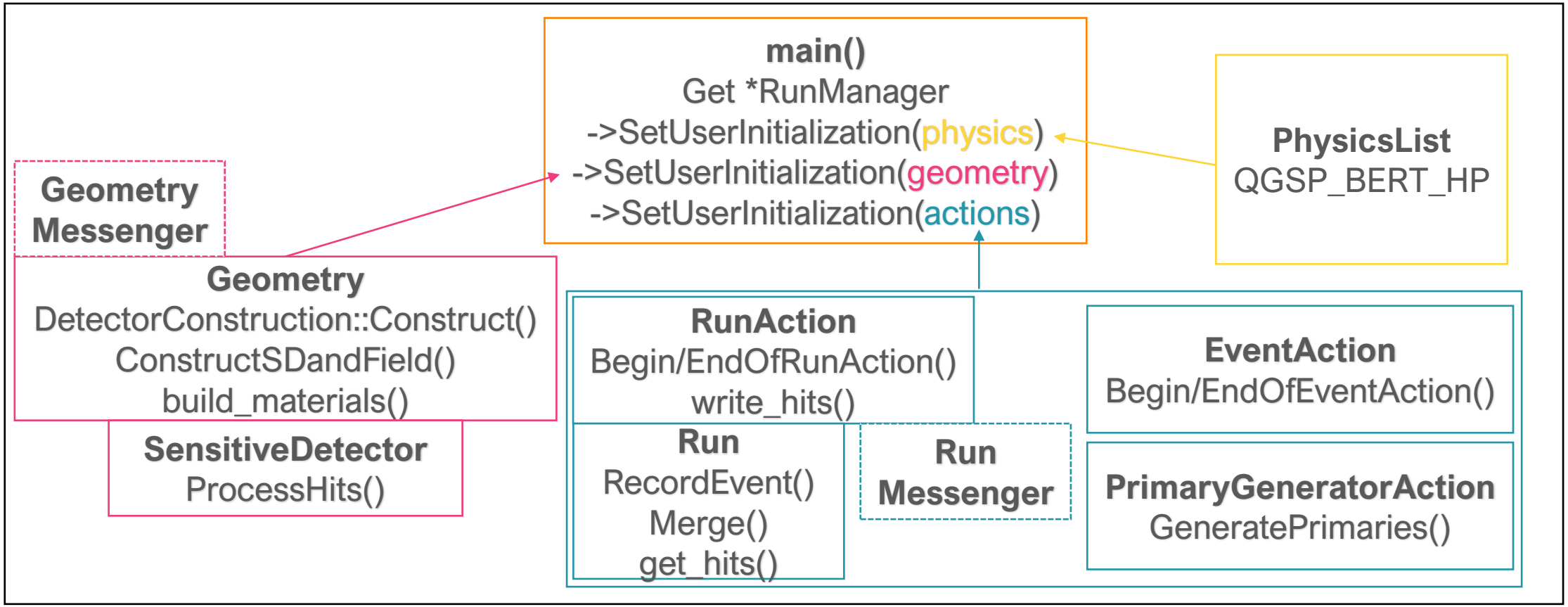
Today's Agenda

- RunMessenger, GeometryMessenger
- Assignment 6 – Due Tonight
- Office hours TODAY, 8-9 PM ET
- Final project formal proposals – due Thursday, midnight PT
  - Any questions?
- Content for remaining lectures

Final Project

- This class's final exam slot is Tuesday, Dec 7, 1:00-3:15 PM ET
- Final project will be due at this time
- Please prepare a 5-minute presentation outlining your work and showing some results

Geant4 Program Anatomy



Geant4: UI Commands

- This is how we modify the simulation parameters without needing to recompile or manually read in config files
- For the command to set the output file path
  - Directory (command path): /ne697/run/save_path
  - Parameter Name(s): save_path
  - Arguments: G4String
  - Default: hits.csv
  - States: G4State_PreInit, G4State_Idle
    - G4State_PreInit: Before /run/initialize
    - G4State_Idle: After /run/initialize, before /run/beamOn or after a run is finished

Geant4: UI Commands

- We’re just defining an interface
- For the command to set the detector size
  - Directory (command): /ne697/geometry/det_size
  - Parameter Name(s): x, y, z
  - Arguments: G4ThreeVector, with units
  - Default: 10x10x10 cm^3 [“10 10 10 cm”]
  - States: G4State_PreInit
    - G4State_PreInit: Before /run/initialize

Geant4: UI Commands

- What you'll do for homework
- For the command to set the detector material
  - Directory (command): /ne697/geometry/det_material
  - Parameter Name: det_material
  - Arguments: G4String
  - Default: G4_SODIUM_IODIDE
  - States: G4State_PreInit
    - G4State_PreInit: Before /run/initialize

Geant4: UI Commands

- Pre-baked commands:
  - G4UIcmdWithoutParameter
  - G4UIcmdWithAString
  - G4UIcmdWithABool
  - G4UIcmdWithAnInteger
  - G4UIcmdWithADouble, G4UIcmdWithADoubleAndUnit
  - G4UIcmdWith3Vector, G4UIcmdWith3VectorAndUnit
- We can define our own, of course!

Geant4: UI Messengers

- Pick a Target (e.g. RunAction; be mindful of multithreading)
- Inherit from G4UImessenger and implement the following:
  - **Constructor**: define/instantiate commands and parameters
  - **Destructor**: clean up the memory (**we** have to do this!)
  - **SetNewValue**(G4UIcommand*, G4String)
    - Check which command was issued, call the Target functions (set_data_path())
    - Geant4 will parse the G4String for us (GetNewDoubleValue, GetNew3VectorValue, GetNewBoolValue, etc)
  - ~~**GetCurrentValue**(G4UIcommand*)~~
    - ~~Check which command was issued, and print the current value~~
    - ~~Usually involves asking the *target* about its current state (file path? Saving data?)~~

Geant4: UI Commands

- General workflow: you need to modify *something* (e.g. the particle source, so PrimaryGeneratorAction) → the Target
  - Create its Messenger if it doesn't exist (inherit from G4UImessenger)
  - Add the UI commands you want
    - /ne697/run/save_path
    - This is the user-facing side of your command
  - Add the getters/setters in the Target to read/write the parameters you want
    - In Messenger, call: RunAction::get_file_path(), DetectorConstruction::set_det_size()
    - In Target, store: G4String: file path, G4ThreeVector: detector size
    - When you go to Construct(), now you've got **m_detSize**
    - This is the code-facing side of your command

Geant4: UI Commands

- What if we want to be able to change the file path and toggle saving data?
- [DEMO]
  - RunMessenger – just need to run!
    - Attached to RunAction, which controls the file IO
    - /ne697/run/save_path, /ne697/run/save_data
    - Takes a string path to a .csv file | Takes a boolean value (true/false)
  - GeometryMessenger - today
    - Attached to DetectorConstruction, which controls the geometry
    - /ne697/geometry/det_size
    - Takes a “3VectorAndUnit” (G4ThreeVector, with units specified)
    - But before we dive back in…

C++: Circular Class Dependencies

- There is a circular dependency between the Messenger and the Target
  - Messenger wants a pointer to Target, so it can call functions on it (RunAction::get_save_path())
  - Target wants a pointer to Messenger, because 1 Target might be made in each thread, and we want each thread to also have a Messenger to propagate messages (and be modified) → Messengers need to be attached to Targets.
  - Who comes first?
- Pointers save us – don't need the Thing itself, just its address
  - What if it's a bad address? Look, they're pointers, not promises

C++ Circular Class Dependencies

- Also, a problem with the headers!
  - runaction.hpp needs runmessenger.hpp, which needs runaction.hpp…uh oh
- Forward declarations mitigate the problem
  - “class RunAction;”
  - “class RunMessenger;”
- We promise that this class exists, we’ll #include it later for the details
- Why don’t we need to know its full structure?
  - Our best friend and worst enemy, ~pointers~!
  - Cannot be stressed enough: *it’s just an address*

Geant4: UI Commands

- [DEMO]
  - GeometryMessenger – as a class
    - Attached to ?, which controls the geometry
    - /ne697/geometry/?
    - Takes a “3VectorAndUnit” (G4?, with units specified)
  - Note: need to add a command for each parameter, of each component
    - This can add up pretty quickly!
  - /ne697/geometry/det_position as a class exercise?
    - Another example doing a similar thing as the size
  - Defining a scintillator?

Remaining Lectures

- Scintillators and very simple optical transport
  - Addition to materials: G4MaterialPropertiesTable
  - Addition to geometry building: optical boundaries (maybe)
- Simple event reconstruction and analysis
  - Ba-133 source next to a scintillator – can we reconstruct the energy spectrum?
- Review and class discussion