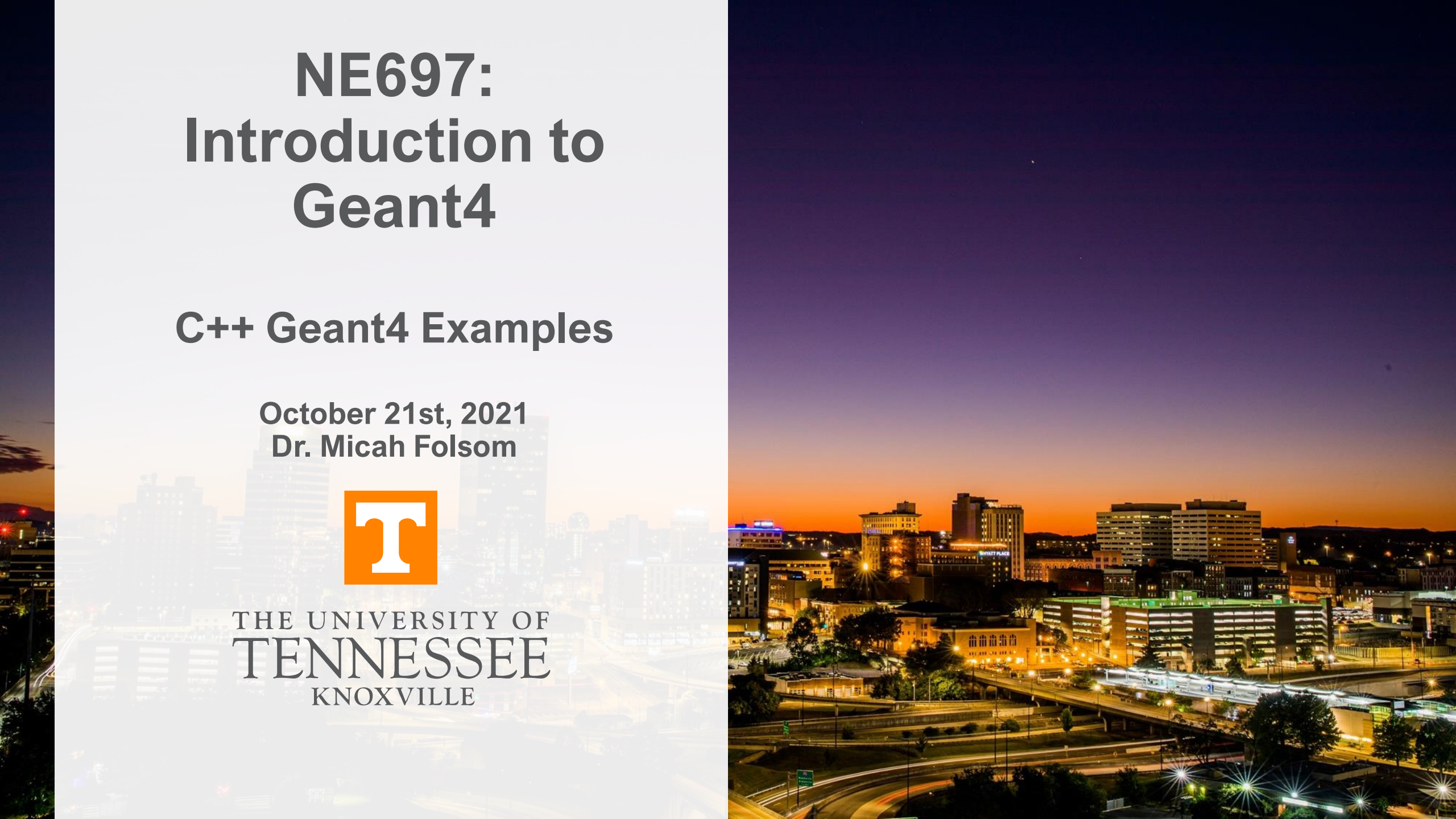
NE697: Introduction to Geant4

C++ Geant4 Examples

October 21st, 2021

Dr. Micah Folsom



THE UNIVERSITY OF TENNESSEE KNOXVILLE

Today's Agenda

- Geant4 core concepts
  - https://geant4-userdoc.web.cern.ch/UsersGuides/ForApplicationDeveloper/html/Fundamentals/fundamentals.html
- Geant4 application anatomy review
- Geant4 examples: B1, B3b, extended/optical/OpNovice
- Start thinking about what you want to do for your final project

Geant4: Run (inherits G4Run)

- “Largest unit of simulation”: a series of G4Events
- Use the Run object to collect information about the G4Events
  - **Run::RecordEvent(G4Event const*),** called after EventAction::EndOfEventAction()
- 1 Run object per thread, +1 for the master (doesn’t do events)
  - **Run::Merge(G4Run const*),** to combine the runs from the different threads at the end
    - Called before **RunAction::EndOfRunAction(G4Run const*)**
    - Merge the G4Run argument into “this” Run object

Geant4: Run

Run::RecordEvent(G4Event const*) called in AnalyzeEvent(currentEvent)

```
00261 void G4RunManager::ProcessOneEvent(G4int i_event)
00262 {
00263   currentEvent = GenerateEvent(i_event);
00264   eventManager->ProcessOneEvent(currentEvent);
00265   AnalyzeEvent(currentEvent);
00266   UpdateScoring();
00267   if(i_event<n_select_msg) G4UImanager::GetUIpointer()->ApplyCommand(msgText);
00268 }
```

```
00292 G4Event* G4RunManager::GenerateEvent(G4int i_event)
00293 {
00294   if(!userPrimaryGeneratorAction)
00295   {
00296     G4Exception("G4RunManager::GenerateEvent()", "Run0032", FatalException,
00297                 "G4VUserPrimaryGeneratorAction is not defined!");
00298     return 0;
00299   }
00300
00301   G4Event* anEvent = new G4Event(i_event);
00302
00303   if(storeRandomNumberStatusToG4Event==1 || storeRandomNumberStatusToG4Event==3)
00304   {
00305     std::ostringstream oss;
00306     HepRandom::saveFullState(oss);
00307     randomNumberStatusForThisEvent = oss.str();
00308     anEvent->SetRandomNumberStatus(randomNumberStatusForThisEvent);
00309   }
00310
00311   if(storeRandomNumberStatus) {
00312     G4String fileN = randomNumberStatusDir + "currentEvent.rndm";
00313     HepRandom::saveEngineStatus(fileN);
00314   }
00315
00316   userPrimaryGeneratorAction->GeneratePrimaries(anEvent);
00317   return anEvent;
00318 }
```

Geant4: G4Event

- Stores all inputs/outputs of an Event
  - Primary particle(s) & vertex(es)
  - Trajectories (we won't use these)
  - Hits
  - Digits (we won't use these)
- Geant4 provides containers for the latter 3, **but we must populate them**
- We will circle back to Hits later, just storing simple information in the Run object for now

Geant4: G4Track, G4Step

- **G4Track**: represents the life of an individual particle
  - 1 G4Track generated for each primary particle in G4Event
  - G4Tracks can spawn more G4Tracks via secondaries
- **G4Step**: 1 step along the G4Track
  - Has boundaries: pre-step point, post-step point
  - Contains information about what happened during that step
- Do not *modify* G4Tracks – you're messing with physics!
- https://geant4-userdoc.web.cern.ch/UsersGuides/ForApplicationDeveloper/html/TrackingAndPhysics/tracking.html

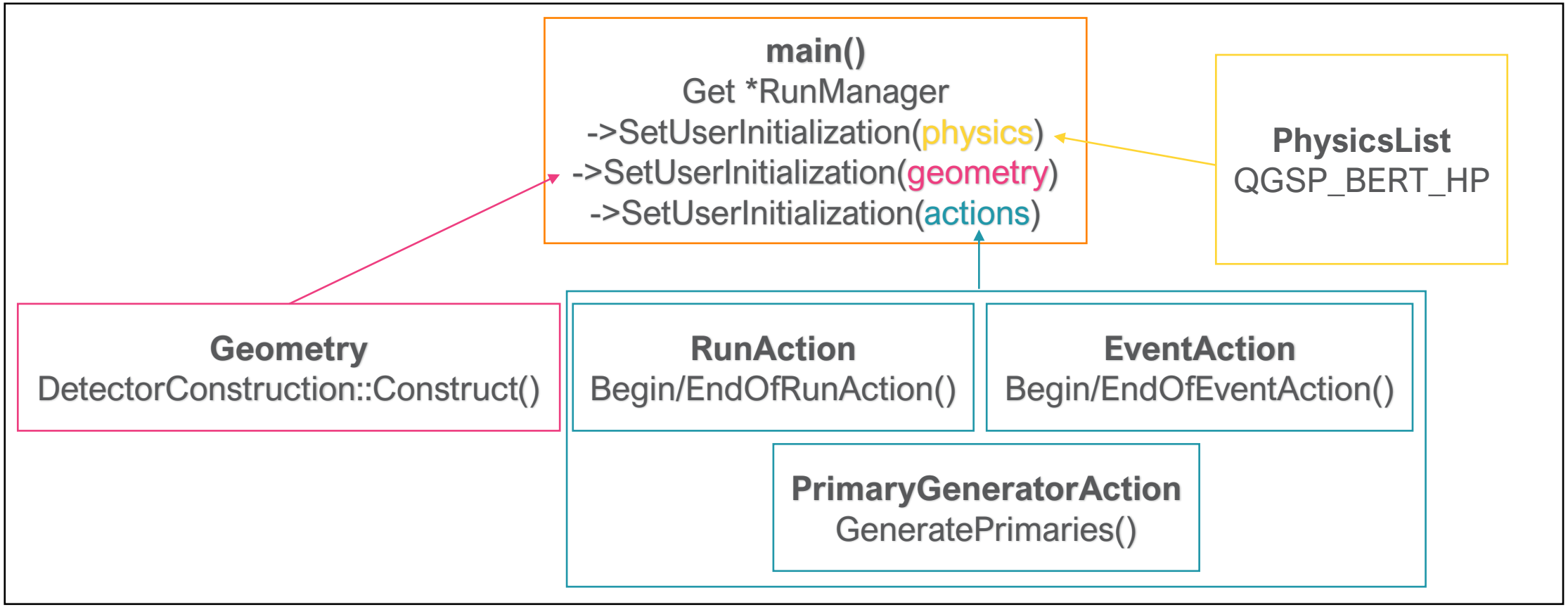
Geant4: Sim Progression

- /run/initialize (calls G4RunManager::Initialize())
- /run/beamOn nevents
  - new Run (1 for each thread), BeginOfRunAction()
- for (int i=0;i < nevents;++i) (split into each thread)
  - new Event, BeginOfEventAction()
  - PrimaryGeneratorAction::GeneratePrimaries()
  - for (int i=0;i < nprimaries;++i)
    - new G4Track → added to the stack
    - G4Step until the track stack is empty
  - EndOfEventAction()
- EndOfRunAction()

Geant4: Application States

- Won’t be using these too much, but just so you’re aware
- **Pre-Init**: before /run/initialize
- **Post-Init**: while /run/initialize is processing
- **Idle**: after /run/initialize, before /run/beamOn
- **GeomClosed**: after /run/beamOn, before the first event
- **EventProc**: while processing events
- **Quit**: while cleanup is taking place (end of program)
- **Abort**: when an exception occurs

Geant4 Core Program Anatomy



Geant4 Geometry

- Capable of building very complicated geometries
  - Parameterized shapes, things with many identical elements (detector pixels)
- We're going to stick with the built-in primitives (Constructed Solid Geometry)
  - https://geant4-userdoc.web.cern.ch/UsersGuides/ForApplicationDeveloper/html/Detector/Geometry/geomSolids.html
- Volumes are nested like MCNP
  - Solid stainless-steel cylinder, then a "solid" air cylinder **inside** of it → air canister
  - Better than defining a hollow cylinder (think about the boundaries)

Geant4 Geometry

- Split into 3 components; build in this order
  - Shape (solid): G4Box, G4Sphere, G4Cons, etc
  - Logical (material): **G4LogicalVolume**, takes pointer to Shape
  - Physical placement (position, rotation, nesting, copying): **G4PVPlacement**, takes pointer to G4LogicalVolume
- Every geometric item must be somewhere inside the outermost volume (world, experimental hall)
  - Returned by DetectorConstruction::Construct()
- Also must define materials (pre-builts in G4NISTManager)
  - G4NISTManager::FindOrBuildMaterial("G4_GALACTIC")

Geant4 Action Classes

- These are the hooks to do things at specific points in the code
  - **RunAction**: Begin, EndOfRunAction()
    - Collect/merge all hit data from the run and write to disk/print to screen
  - **EventAction**: Begin, EndOfEventAction
    - Collect hits from the event history (1 source particle, 1 decay, etc)
  - **SteppingAction**: UserSteppingAction()
    - Generate hits when certain criteria are met (e.g. energy deposited > 0)
  - TrackingAction: Pre, PostUserTrackingAction()
  - StackingAction: ClassifyNewTrack()
  - **PrimaryGeneratorAction**: GeneratePrimaries()
    - Set the primary particle(s') properties (may be multiple, e.g. Cf-252)
    - Can be an ion of an isotope that decays

Geant4 Units

- In .cpp, #include “G4SystemOfUnits.hh”
  - Do **not** include it in headers (.hpp files)
- Multiply to apply units
  - float world_x = 5*m;
  - (G4float world_x = 5*m;)
  - float det_y = 10*cm;
  - float particle_energy = 661.7*keV;
- Divide by desired unit to remove units
  - world_x_cm = world_x / cm;
  - det_y = det_y / um;
- G4cout << G4BestUnit(det_y, “Length”) << G4endl;

Lab Time: Geant4 Examples

- B1: dose calculations
  - Modify B1 to output periodic updates of the event number being processed
    - BeginOfEventAction() or EndOfEventAction()
      - Get the G4RunManager, 2. Get the current Run object, 3. Get the total # of events
  - Create .mac files that run with 100,000 particles for these source configurations
    - Gamma, 511 keV
    - Neutron, 2 MeV
    - e-, 1 MeV
  - Change the size of the phantom objects, recompile and confirm they changed
  - Modify the geometry to use a G4Sphere instead of a G4Cons (use a similar size)
  - Add another analysis metric that gets printed to the screen: eDep^3

Lab Time: Geant4 Examples

- B3b: PET scanner system
  - Uses a hook we haven't discussed: Stacking Action
    - Check out B3StackingAction
    - ClassifyNewTrack() triggered when a G4Track is created
    - Gives the opportunity to kill particles we don't care about (secondaries)
  - Run with 100,000 particles, and count the "Nb of good e+ annihilations"
  - Change the material to NaI instead of Lu2SiO5
  - Run again with 100,000 particles – how does it compare?
  - Check out B3bRun::RecordEvent
    - What is the energy threshold?
  - Change the source to N-13 with the /gun/particle command (see run2.mac for help) and run again with 100,000 particles

Lab Time: Geant4 Examples

- OpNovice (extended/optical/OpNovice): simple optical photon example
- [DEMO]
  - Material building in DetectorConstruction::Construct()
  - Optical properties for materials and boundaries
  - Scintillation properties
  - Checking the process in StackingAction
  - Custom UI command in PrimaryGeneratorMessenger