Review of Geant4 Hands-on sessions

Department of Physics, Yonsei Univ.

Jaeyoung Kim (jaeyoung_kim@yonsei.ac.kr)

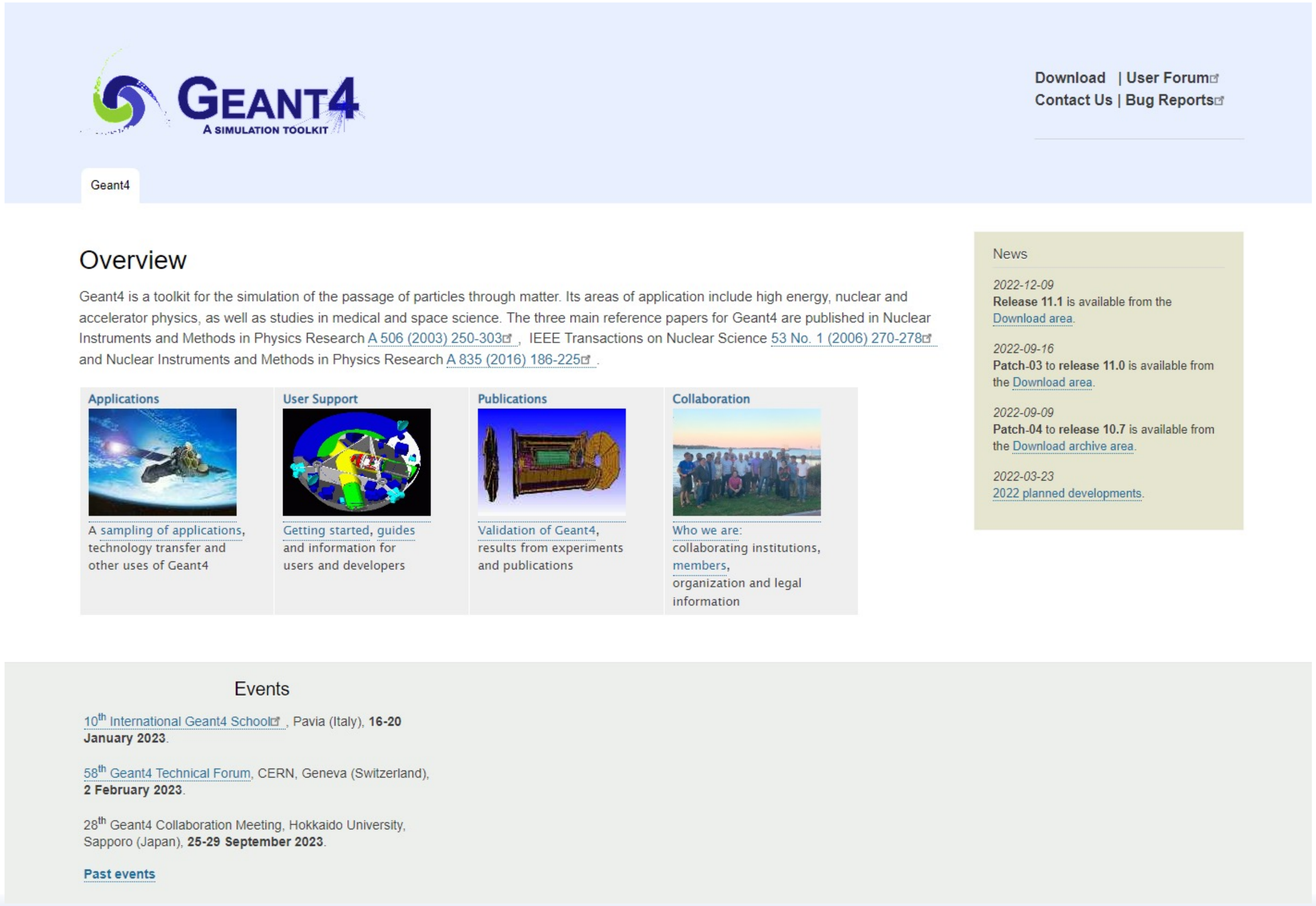
2022.12.27.

Contents

- ✦ Documentations
- ✦ Hands-on sessions

Website

✦ Geant4 official site: https://geant4.org/



Download | User Forum
Contact Us | Bug Reports

Geant4

Overview

Geant4 is a toolkit for the simulation of the passage of particles through matter. Its areas of application include high energy, nuclear and accelerator physics, as well as studies in medical and space science. The three main reference papers for Geant4 are published in Nuclear Instruments and Methods in Physics Research A 506 (2003) 250-303 , IEEE Transactions on Nuclear Science 53 No. 1 (2006) 270-278 and Nuclear Instruments and Methods in Physics Research A 835 (2016) 186-225 .

Applications
A sampling of applications, technology transfer and other uses of Geant4

User Support
Getting started, guides and information for users and developers

Publications
Validation of Geant4, results from experiments and publications

Collaboration
Who we are: collaborating institutions, members, organization and legal information

News

2022-12-09
Release 11.1 is available from the Download area.

2022-09-16
Patch-03 to **release 11.0** is available from the Download area.

2022-09-09
Patch-04 to **release 10.7** is available from the Download archive area.

2022-03-23
2022 planned developments.

Events

10th International Geant4 School , Pavia (Italy), **16-20 January 2023**.

58th Geant4 Technical Forum, CERN, Geneva (Switzerland), **2 February 2023**.

28th Geant4 Collaboration Meeting, Hokkaido University, Sapporo (Japan), **25-29 September 2023**.

Past events

Documentations

The following documents are accessible through the Geant4 website

✦ **Book for Application Developers** [link]: **introduces the first-time user to Geant4**, provides a description of the available tools and supply the practical information required to develop and run simulation applications

✦ **Physics Reference Manual** [link]: **presents the theoretical formulation, model, or parameterization of the physics interaction**s and describes the probability of the occurrence of an interaction and the sampling mechanisms required to simulate it

✦ **Users Guide for Toolkit Developers** [link]: provides information for those who want to understand or refer to the **detailed design of the toolkit**, as well as procedures for extending the functionality of the toolkit

Hands-on sessions

- ✦ Hands-on 1: **installation**, run a **basic example**
- ✦ Hands-on 2: **defining materials**, a very simple geometry
- ✦ Hands-on 3: **building geometry**, sensitive detectors and hits, user actions
- ✦ Hands-on 4: using g4analysis **to store the results in a file**

All materials and manuals are uploaded in indico.

Hands-on 1

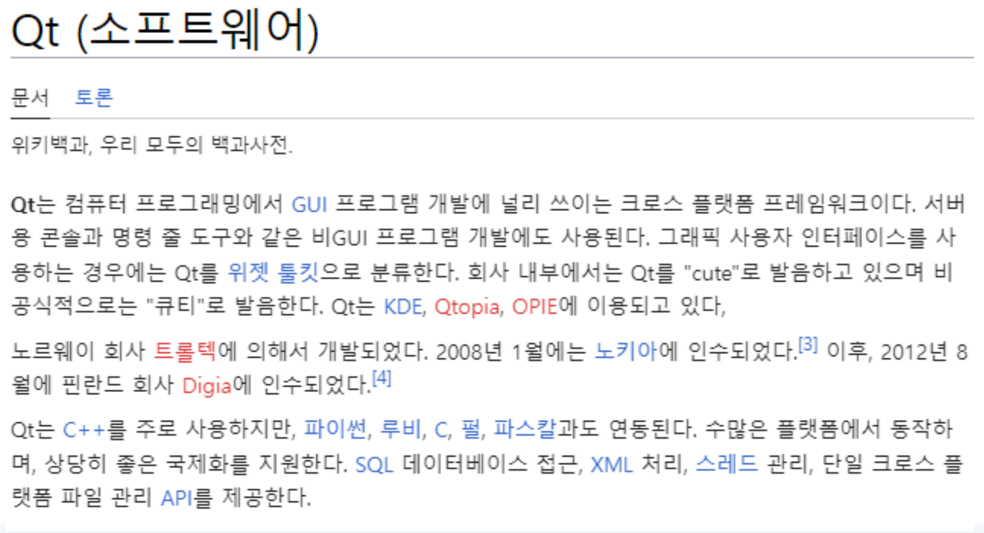
Hands-on 1: installation

✦ #1: compile

1. Download source codes (https://geant4.web.cern.ch/support/download)
2. go to the path and unzip it.
3. `mkdir build`
4. `cd build`
5. Path to install
   ```
   cmake -DCMAKE_INSTALL_PREFIX=~/geant4-v11.1.0-install -DGEANT4_INSTALL_DATA=ON -DGEANT4_USE_OPENGL_X11=ON -DGEANT4_USE_QT=ON ~/geant4-v11.1.0
   ```
   Location of source codes and data
6. `make -j20`
7. `make install -j20`

✦ #2: Set environment variable

1. `source ~/geant4-v11.1.0-install/bin/geant4.sh`



Qt (소프트웨어)

문서 토론

위키백과, 우리 모두의 백과사전.

Qt는 컴퓨터 프로그래밍에서 GUI 프로그램 개발에 널리 쓰이는 크로스 플랫폼 프레임워크이다. 서버용 콘솔과 명령 줄 도구와 같은 비GUI 프로그램 개발에도 사용된다. 그래픽 사용자 인터페이스를 사용하는 경우에는 Qt를 위젯 툴킷으로 분류한다. 회사 내부에서는 Qt를 "cute"로 발음하고 있으며 비공식적으로는 "큐티"로 발음한다. Qt는 KDE, Qtopia, OPIE에 이용되고 있다,

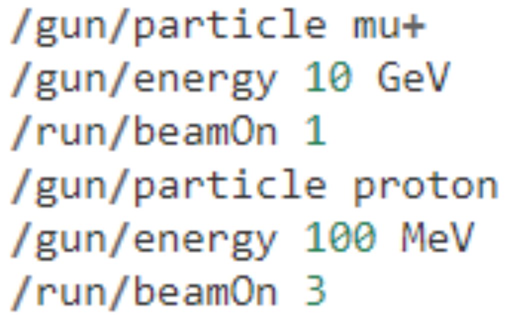
노르웨이 회사 트롤텍에 의해서 개발되었다. 2008년 1월에는 노키아에 인수되었다.[3] 이후, 2012년 8월에 핀란드 회사 Digia에 인수되었다.[4]

Qt는 C++를 주로 사용하지만, 파이썬, 루비, C, 펄, 파스칼과도 연동된다. 수많은 플랫폼에서 동작하며, 상당히 좋은 국제화를 지원한다. SQL 데이터베이스 접근, XML 처리, 스레드 관리, 단일 크로스 플랫폼 파일 관리 API를 제공한다.

Hands-on 1: run a basic example

✦ Steps

1. `cd ~/geant4-v11.1.0`
2. `cd examples/basic/B1`
3. `mkdir bulid`
4. `cd build`
5. `cmake ../`
6. `make`
7. `./exampleB1`
8. `/run/beamOn 100`

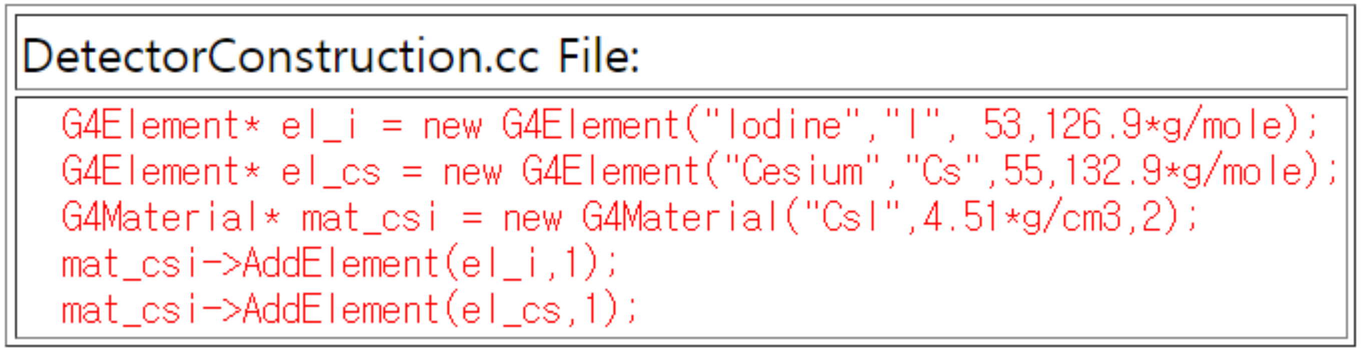


```
/gun/particle mu+
/gun/energy 10 GeV
/run/beamOn 1
/gun/particle proton
/gun/energy 100 MeV
/run/beamOn 3
```

Hands-on 2

Hands-on 2: defining materials

- For Cs: Z=55 , A_{eff}=132.9g/mol
- For I : Z=53 , A_{eff} =126.9g/mol
- Density of crystal of CsI: ρ=4.51g/cm^3



DetectorConstruction.cc File:

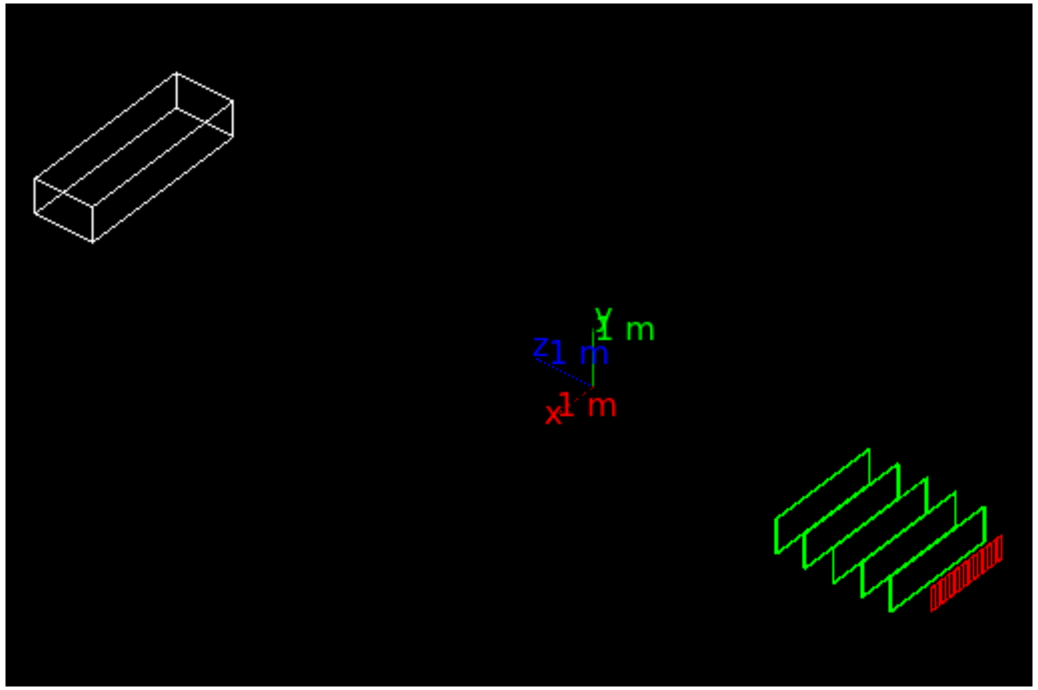
```
G4Element* el_i = new G4Element("Iodine","I", 53,126.9*g/mole);
G4Element* el_cs = new G4Element("Cesium","Cs",55,132.9*g/mole);
G4Material* mat_csi = new G4Material("CsI",4.51*g/cm3,2);
mat_csi->AddElement(el_i,1);
mat_csi->AddElement(el_cs,1);
```

Hands-on 2:a simple geometry

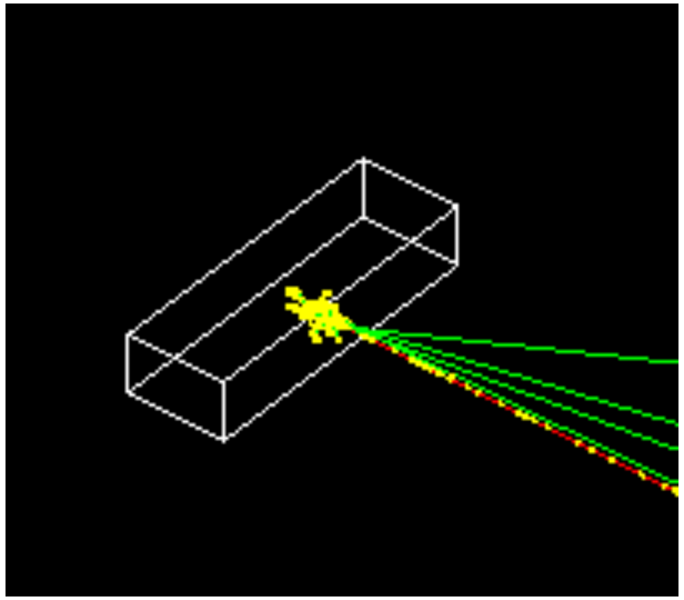
✦ Construct a geometry

DetectorConstruction.cc File:

```
G4Material* material = G4Material::GetMaterial("CsI");
G4VSolid* hadCalorimeterSolid = new G4Box("HadCalorimeterBox",1.5*m,30.*cm,50.*cm);
G4LogicalVolume* hadCalorimeterLogical = new G4LogicalVolume(hadCalorimeterSolid,material,"HadCalorimeterLogical");
new G4PVPlacement(0,G4ThreeVector(0.,0.,3.*m),hadCalorimeterLogical,"HadCalorimeterPhysical",secondArmLogical,false,0,checkOverlaps);
```



/run/beamOn 1



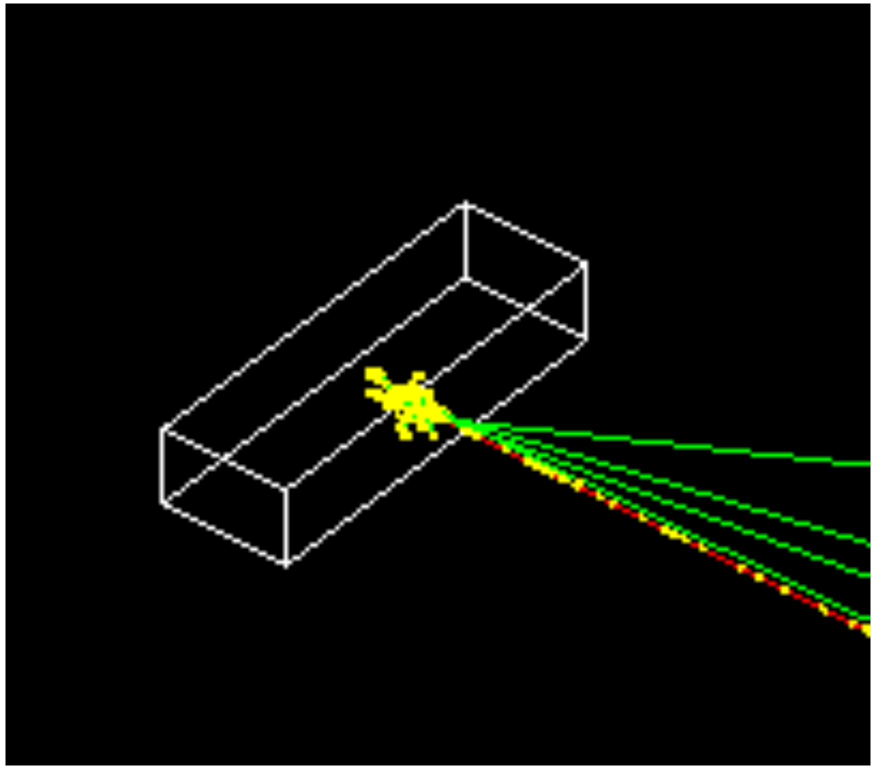
Hands-on 2: change materials

✦ Difference: CsI → Scintillator

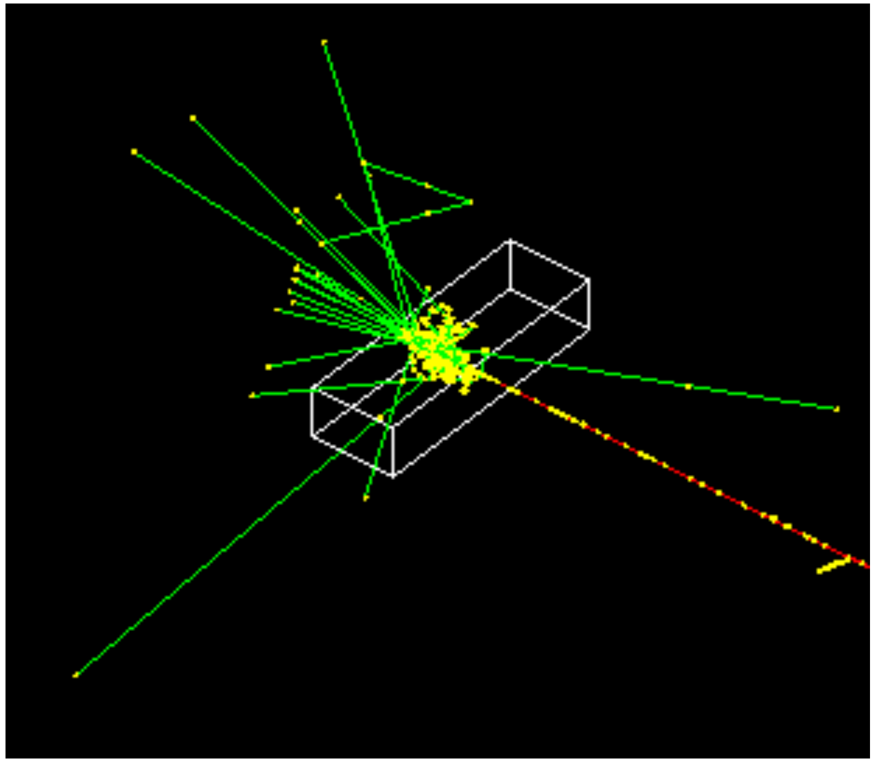
Then,

```
/gun/particle e-
/run/beamOn 1
```

CsI



Scintillator



Hands-on 3

Hands-on 3: building geometry

✦ Steps

1. Build second hodoscope

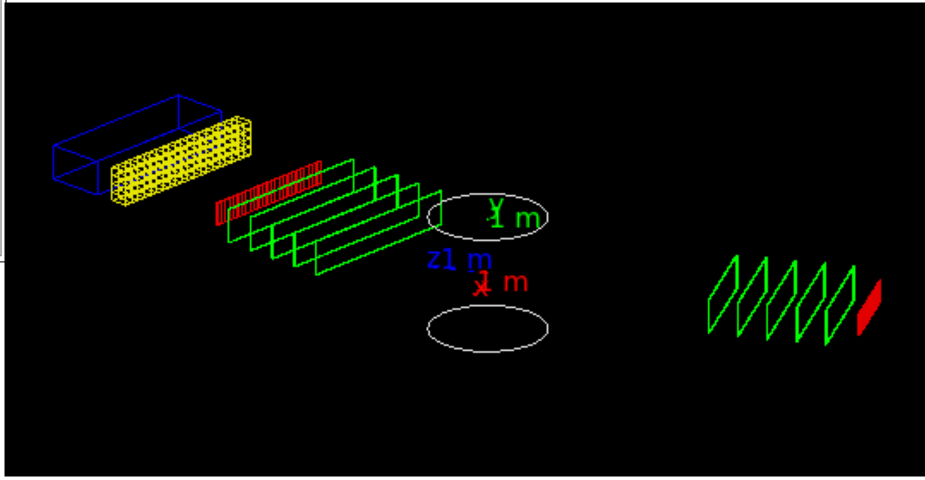
DetectorConstruction.cc File:

```
// ============================================
// Exercise 1
// Complete the full geometry.
// Note that second arm, by default is rotated of
// 30 deg.
// Step 1: Add an hodoscope with dimensions (X,Y,Z):
// (10,40,1)cm made of scintillator.
// There are 25 planes placed at Y=Z=0 (w.r.t. mother volume)
// hodoscopes in second arm
G4VSolid* hodoscope2Solid = new G4Box("hodoscope2Box",5.*cm,20.*cm,0.5*cm);
fHodoscope2Logical = new G4LogicalVolume(hodoscope2Solid,scintillator,"hodoscope2Logical");
for (G4int i=0;i<25;i++)
{
  G4double x2 = (i-12)*10.*cm;
  new G4PVPlacement(0,G4ThreeVector(x2,0.,0.),fHodoscope2Logical,"hodoscope2Physical",secondArmLogical,false,i,checkOverlaps);
}
```

2. Build drift chambers

DetectorConstruction.cc File:

```
// Step 2: Add 5 drift chambers made of argon, with dimensions (X,Y,Z):
// (300,60,2)cm
// These are placed equidistant inside the second arm at distances from -2.5m
// to -0.5m
// drift chambers in second arm
G4VSolid* chamber2Solid = new G4Box("chamber2Box",1.5*m,30.*cm,1.*cm);
G4LogicalVolume* chamber2Logical = new G4LogicalVolume(chamber2Solid,argonGas,"chamber2Logical");
for (G4int i=0;i<5;i++)
{
  G4double z2 = (i-2)*0.5*m - 1.5*m;
  new G4PVPlacement(0,G4ThreeVector(0.,0.,z2),chamber2Logical,"chamber2Physical",secondArmLogical,false,i,checkOverlaps);
}
```



Hands-on 3: building geometry

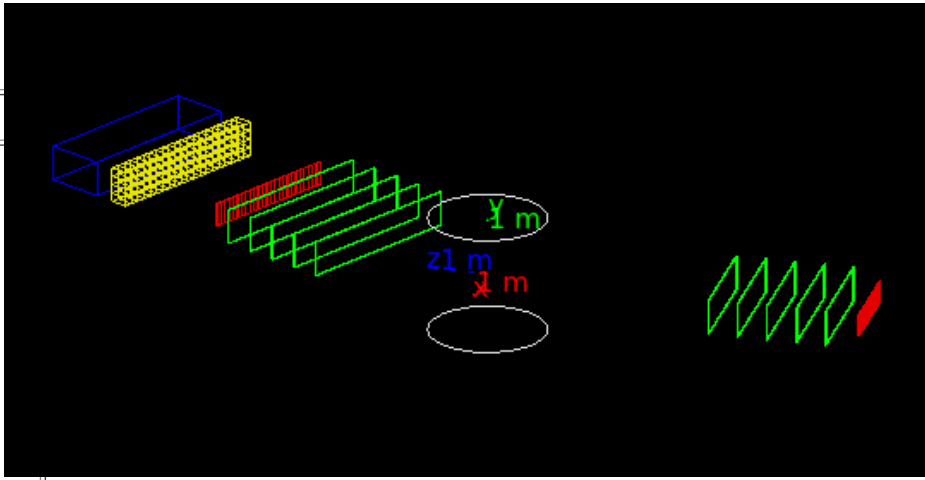
✦ Steps

3. Add a virtual wire plane in drift chambers

DetectorConstruction.cc File:

```
// Step 3: Add a virtual wire plane of (300,60,0.02)cm
// at (0,0,0) in the drift chamber
// virtual wire plane
G4VSolid* wirePlane2Solid = new G4Box("wirePlane2Box",1.5*m,30.*cm,0.1*mm);
fWirePlane2Logical = new G4LogicalVolume(wirePlane2Solid,argonGas,"wirePlane2Logical");
new G4PVPlacement(0,G4ThreeVector(0.,0.,0.),fWirePlane2Logical, "wirePlane2Physical",chamber2Logical,false,0,checkOverlaps);
```

4. Build an EM calorimeter



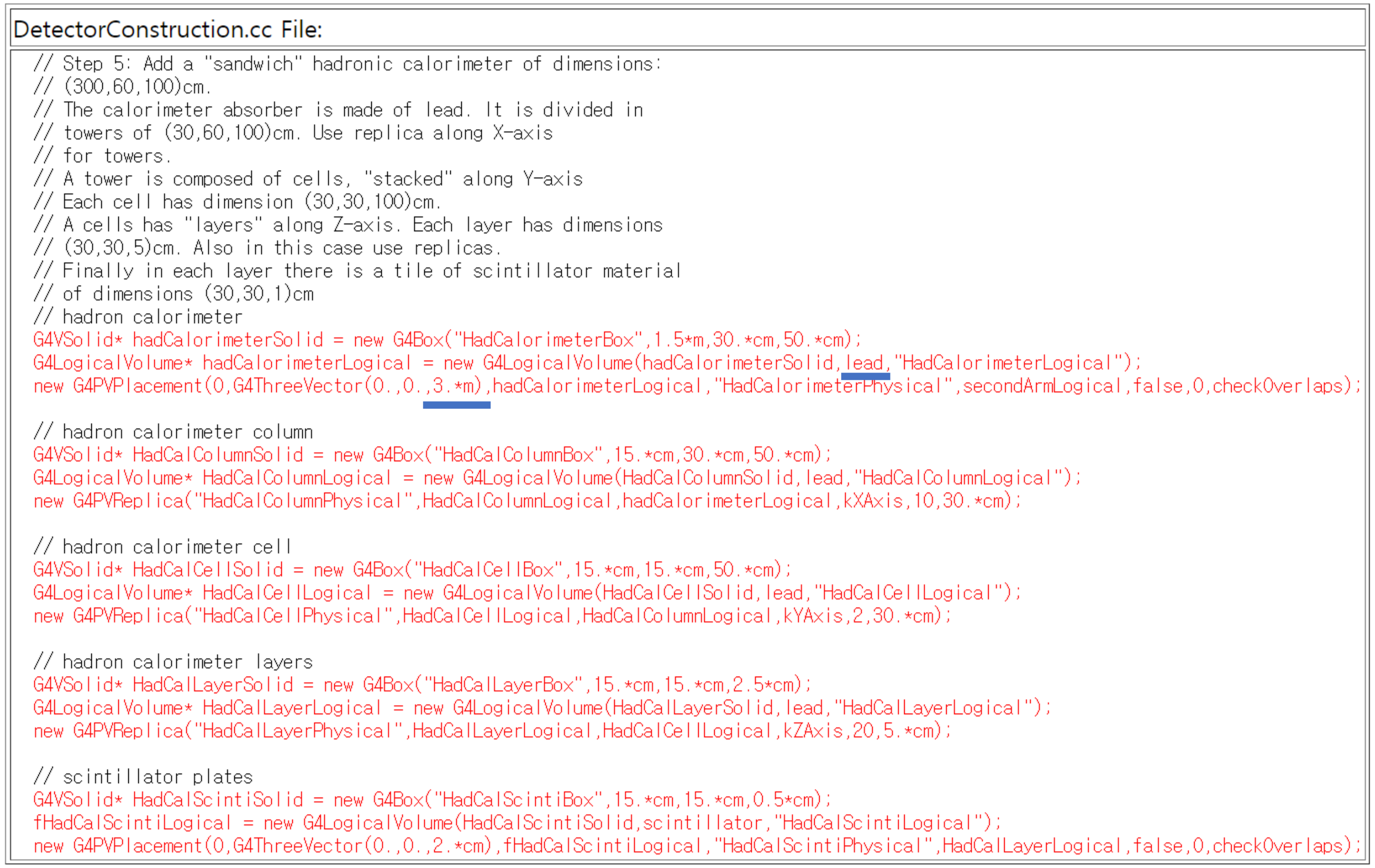
DetectorConstruction.cc File:

```
// Step 4: Build CsI EM-calorimeter of (300,60,30)cm
// placed at (0,0,2)m in the second arm.
// The calorimeter is made of 80 cells,
// parametrised according to CellParametrisation
// G4VPVParameterisation concrete instance.
// This class parametrize the position of each cell depending
// on its copy number.
// The cells have dimensions 15x15x30 cm.
// (you could use placements or replicas, but here
// we show how to use parametrisations to build geometry)
// CsI calorimeter
G4VSolid* emCalorimeterSolid = new G4Box("EMcalorimeterBox",1.5*m,30.*cm,15.*cm);
G4LogicalVolume* emCalorimeterLogical = new G4LogicalVolume(emCalorimeterSolid,csI,"EMcalorimeterLogical");
new G4PVPlacement(0,G4ThreeVector(0.,0.,2.*m),emCalorimeterLogical,"EMcalorimeterPhysical",secondArmLogical,false,0,checkOverlaps);

// EMcalorimeter cells
G4VSolid* cellSolid = new G4Box("cellBox",7.5*cm,7.5*cm,15.*cm);
fCellLogical = new G4LogicalVolume(cellSolid,csI,"cellLogical");
G4VPVParameterisation* cellParam = new CellParameterisation();
new G4PVParameterised("cellPhysical",fCellLogical,emCalorimeterLogical,kXAxis,80,cellParam);
```

Hands-on 3: building geometry

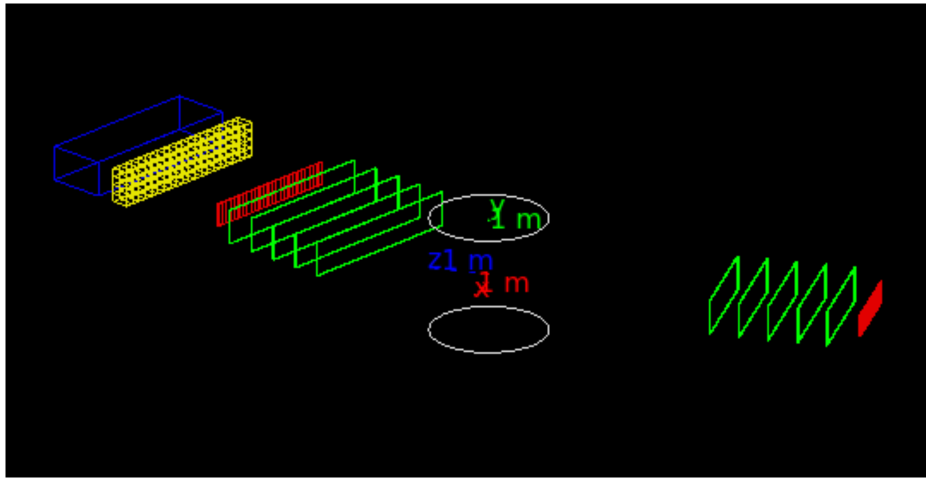
✦ Steps

5. Implement hadronic calorimeter



DetectorConstruction.cc File:

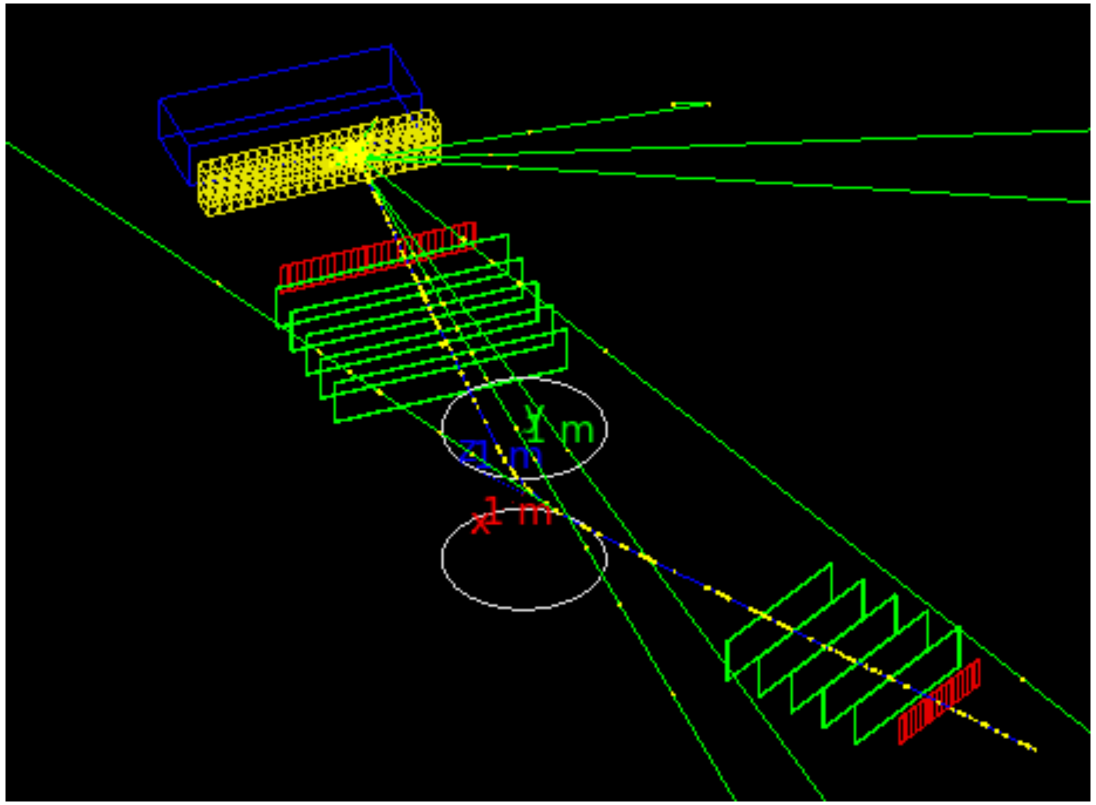
```
// Step 5: Add a "sandwich" hadronic calorimeter of dimensions:
// (300,60,100)cm.
// The calorimeter absorber is made of lead. It is divided in
// towers of (30,60,100)cm. Use replica along X-axis
// for towers.
// A tower is composed of cells, "stacked" along Y-axis
// Each cell has dimension (30,30,100)cm.
// A cells has "layers" along Z-axis. Each layer has dimensions
// (30,30,5)cm. Also in this case use replicas.
// Finally in each layer there is a tile of scintillator material
// of dimensions (30,30,1)cm
// hadron calorimeter
G4VSolid* hadCalorimeterSolid = new G4Box("HadCalorimeterBox",1.5*m,30.*cm,50.*cm);
G4LogicalVolume* hadCalorimeterLogical = new G4LogicalVolume(hadCalorimeterSolid,lead,"HadCalorimeterLogical");
new G4PVPlacement(0,G4ThreeVector(0.,0.,3.*m),hadCalorimeterLogical,"HadCalorimeterPhysical",secondArmLogical,false,0,checkOverlaps);

// hadron calorimeter column
G4VSolid* HadCalColumnSolid = new G4Box("HadCalColumnBox",15.*cm,30.*cm,50.*cm);
G4LogicalVolume* HadCalColumnLogical = new G4LogicalVolume(HadCalColumnSolid,lead,"HadCalColumnLogical");
new G4PVReplica("HadCalColumnPhysical",HadCalColumnLogical,hadCalorimeterLogical,kXAxis,10,30.*cm);

// hadron calorimeter cell
G4VSolid* HadCalCellSolid = new G4Box("HadCalCellBox",15.*cm,15.*cm,50.*cm);
G4LogicalVolume* HadCalCellLogical = new G4LogicalVolume(HadCalCellSolid,lead,"HadCalCellLogical");
new G4PVReplica("HadCalCellPhysical",HadCalCellLogical,HadCalColumnLogical,kYAxis,2,30.*cm);

// hadron calorimeter layers
G4VSolid* HadCalLayerSolid = new G4Box("HadCalLayerBox",15.*cm,15.*cm,2.5*cm);
G4LogicalVolume* HadCalLayerLogical = new G4LogicalVolume(HadCalLayerSolid,lead,"HadCalLayerLogical");
new G4PVReplica("HadCalLayerPhysical",HadCalLayerLogical,HadCalCellLogical,kZAxis,20,5.*cm);

// scintillator plates
G4VSolid* HadCalScintiSolid = new G4Box("HadCalScintiBox",15.*cm,15.*cm,0.5*cm);
fHadCalScintiLogical = new G4LogicalVolume(HadCalScintiSolid,scintillator,"HadCalScintiLogical");
new G4PVPlacement(0,G4ThreeVector(0.,0.,2.*cm),fHadCalScintiLogical,"HadCalScintiPhysical",HadCalLayerLogical,false,0,checkOverlaps);
```



Hands-on 3: user actions

✦ Command-line user actions:

```
Then,
/gun/particle e+
/run/beamOn 1
```



+y axis: 1T magnetic field

✦ Can change with the user-defined command:
Ex: `/tutorial/field/value 0.5 tesla`

✦ Can change the second arm’s angle:
Ex: `/tutorial/detector/armAngle 60`

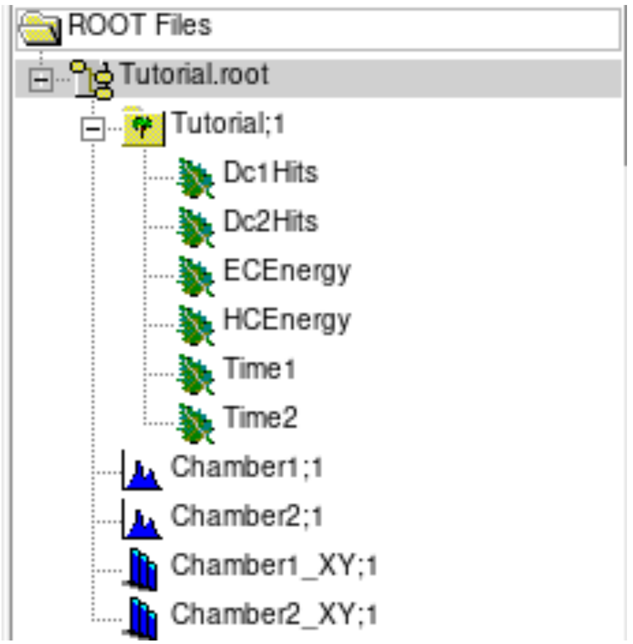
Hands-on 4

Hands-on 4: Physics measurements

✦ Using ‘**G4AnalaysisManager**’, store ntuples and histograms (csv, hdf5, root,xml are available.)

RunAction.cc file:

```
RunAction::RunAction()
 : G4UserRunAction()
{
 // Create analysis manager
 // Create analysis manager, set output format and file name
 auto analysisManager = G4AnalysisManager::Instance();

 // Default settings
 analysisManager->SetDefaultFileType("root");
 G4cout << "Using " << analysisManager->GetType() << G4endl;
 analysisManager->SetVerboseLevel(1);
 analysisManager->SetFileName("Tutorial");

 // Book histograms, ntuple
 //

 // Creating 1D histograms
 analysisManager->CreateH1("Chamber1","Drift Chamber 1 # Hits", 50, 0., 50); // h1 Id = 0
 analysisManager->CreateH1("Chamber2","Drift Chamber 2 # Hits", 50, 0., 50); // h1 Id = 1

 // Creating 2D histograms
 analysisManager->CreateH2("Chamber1_XY","Drift Chamber 1 X vs Y",50, -1000., 1000, 50, -300., 300.); // h2 Id = 0
 analysisManager->CreateH2("Chamber2_XY","Drift Chamber 2 X vs Y",50, -1500., 1500, 50, -300., 300.); // h2 Id = 1

 // Creating ntuple
 //
 analysisManager->CreateNtuple("Tutorial", "Hits");
 analysisManager->CreateNtupleIColumn("Dc1Hits"); // column Id = 0
 analysisManager->CreateNtupleIColumn("Dc2Hits"); // column Id = 1
 analysisManager->CreateNtupleDColumn("ECEnergy"); // column Id = 2
 analysisManager->CreateNtupleDColumn("HCEnergy"); // column Id = 3
 analysisManager->CreateNtupleDColumn("Time1"); // column Id = 4
 analysisManager->CreateNtupleDColumn("Time2"); // column Id = 5
 analysisManager->FinishNtuple(); //Do not forget this line!
}
```



Available in test4 server

✦ Already compiled in our test4 server.

✦ **You can play with Geant4 freely**.
Complied directory: /home/jykim/geant4-v11.1.0-install
Data directory: /home/jykm/geant4-v11.1.0

✦ **All materials and manuals(.pdf)** are uploaded in indico.

Available in test4 server

✦ Already compiled in our test4 server.

✦ You can play with Geant4 freely.

Complied directory: /home/jykim/geant4-v11.1.0-install

Data directory: /home/jykm/geant4-v11.1.0