Introduction to Geant4

Yeonju Go

2014 01 07 Tue.

What is Geant4?

- GEometry ANd Traking : A Monte Carlo software toolkit to simulate the passage of particles through matter

- It is for ...
  - detector simulation of research in high energy physics,
    nuclear physics,
    cosmic ray physics.
  - application : space science, radiological science, etc.

How can we apply this?

- We can input initial conditions.
  - projectile : id, energy, charge, and so on.
  - geometry of detector : shape, material, and so on.

- The program automatically transport the particle by simulating particle interactions in matter.

Grammar of Geant4

- based on c++

- header file : ~.hh
- source file : ~.cc
- class name and basic variables name : start with 'G4~'
  - ex) G4Step, G4LogicalVolume, G4int, G4double ...
- distinguish the word by writing with large letter.
  - ex) G4UserAction, G4PhysicalVolume::GetName()

Unit System

- each physics quantity must be multiplied by its proper unit.
    - radius = 10.0 * cm;
    - kineticE = 1.0 * GeV;
- mm, ns, MeV, eplus(positron charge), kelvin, mole, candela(Luminosity intensity), radian, steradian ... and so on.

Examples and References

- in KU server,
  - cp -r $G4INSTALL/examples/novice [your directory]
- GEANT4 무작정 따라하기 by Genie.
- http://geant4.cern.ch/support/index.shtml
  - Geant4 User's guide.pdf
  - Geant4 PhysicsReferenceManual.pdf
  - ... and so on

Let's make!

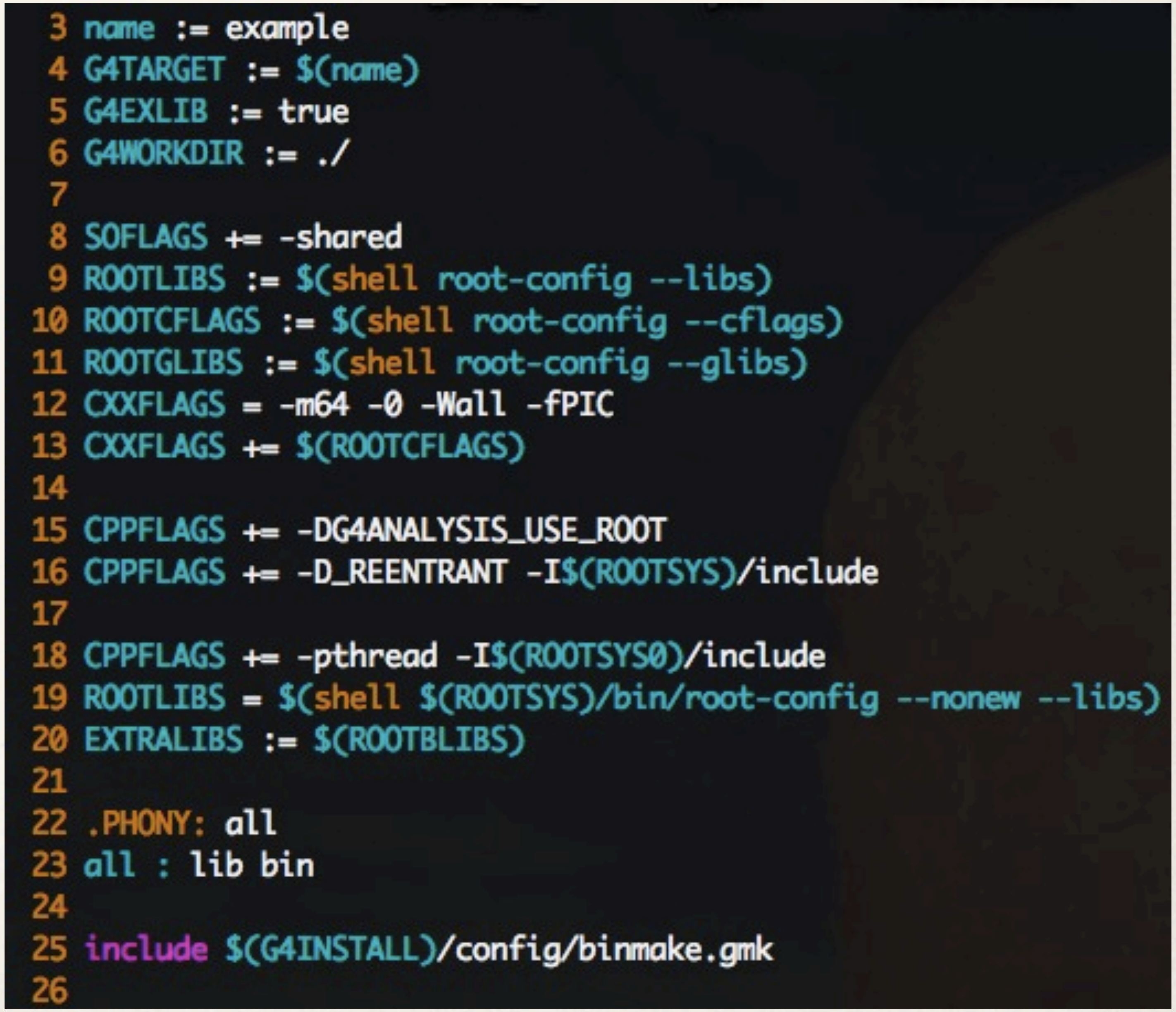
- based on 'GEANT4 무작정 따라하기'

List of most basic files

- main directory
  - GNUmakefile
  - example.cc
  - "include" directory
    - examplePhysicsList.hh
    - exampleDetectorConstruction.hh
    - examplePrimaryGeneratorAction.hh
  - "src" directory
    - examplePhysicsList.cc
    - exampleDetectorConstruction.cc
    - examplePrimaryGeneratorAction.cc

GNUmakefile

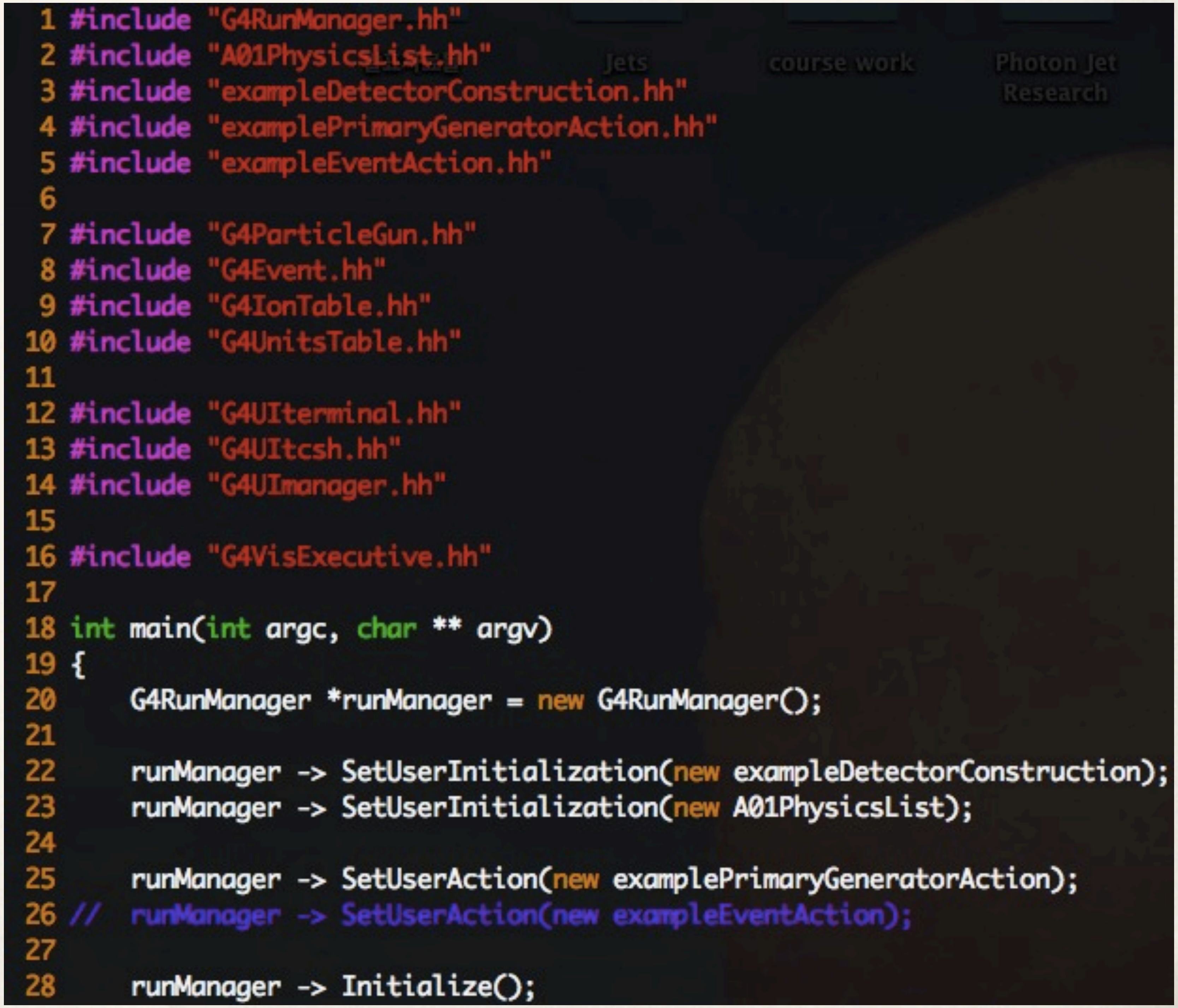
- name := [name of ~.cc file]
- line 8~20 : about root library



```
 3 name := example
 4 G4TARGET := $(name)
 5 G4EXLIB := true
 6 G4WORKDIR := ./
 7
 8 SOFLAGS += -shared
 9 ROOTLIBS := $(shell root-config --libs)
10 ROOTCFLAGS := $(shell root-config --cflags)
11 ROOTGLIBS := $(shell root-config --glibs)
12 CXXFLAGS = -m64 -0 -Wall -fPIC
13 CXXFLAGS += $(ROOTCFLAGS)
14
15 CPPFLAGS += -DG4ANALYSIS_USE_ROOT
16 CPPFLAGS += -D_REENTRANT -I$(ROOTSYS)/include
17
18 CPPFLAGS += -pthread -I$(ROOTSYS0)/include
19 ROOTLIBS = $(shell $(ROOTSYS)/bin/root-config --nonew --libs)
20 EXTRALIBS := $(ROOTBLIBS)
21
22 .PHONY: all
23 all : lib bin
24
25 include $(G4INSTALL)/config/binmake.gmk
26
```

example.cc

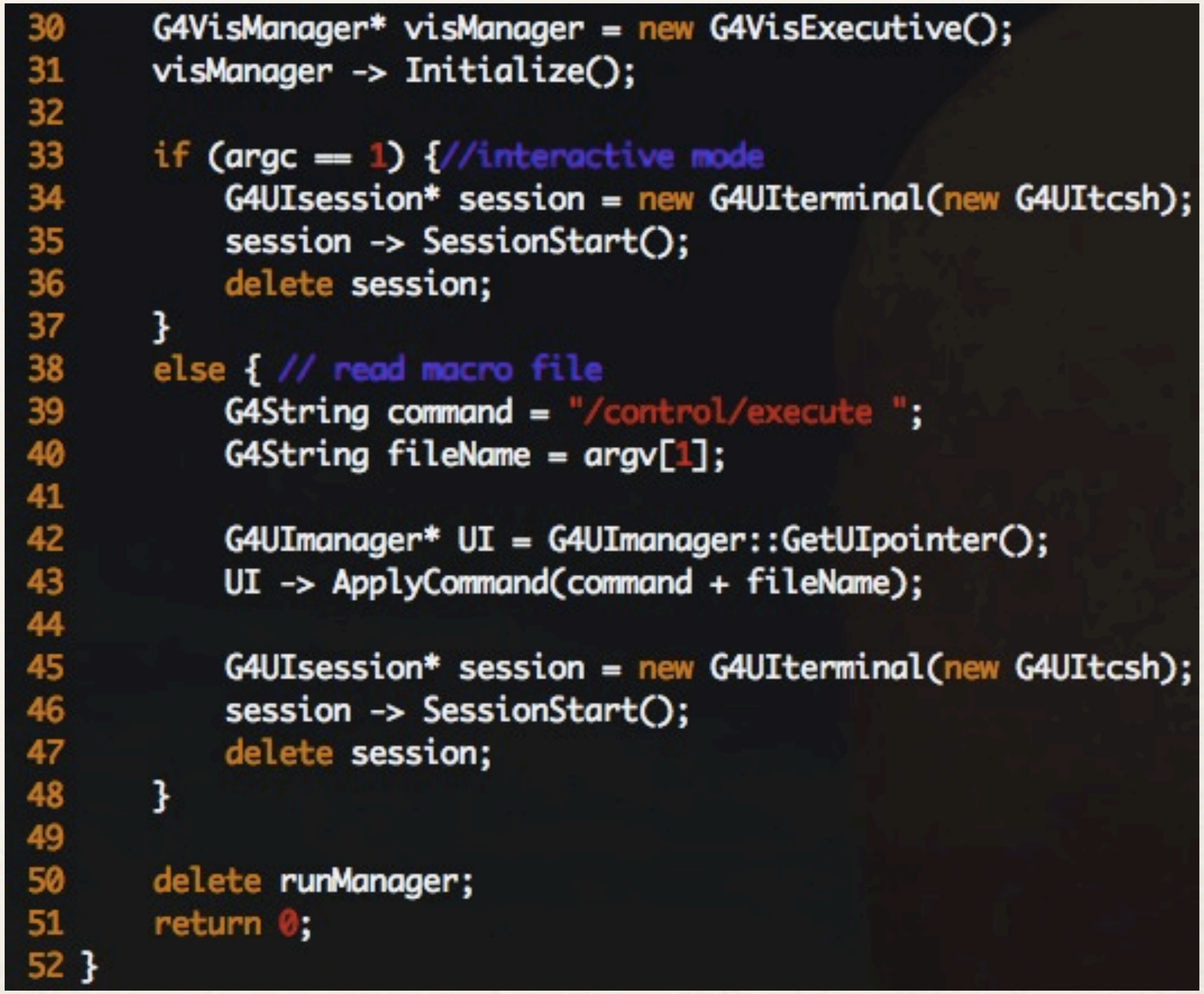
- G4RunManager
  - Set initialization (detector, environment)
  - Set Action (beam information)
  - BeamOn(1);



```
#include "G4RunManager.hh"
#include "A01PhysicsList.hh"
#include "exampleDetectorConstruction.hh"
#include "examplePrimaryGeneratorAction.hh"
#include "exampleEventAction.hh"

#include "G4ParticleGun.hh"
#include "G4Event.hh"
#include "G4IonTable.hh"
#include "G4UnitsTable.hh"

#include "G4UIterminal.hh"
#include "G4UItcsh.hh"
#include "G4UImanager.hh"

#include "G4VisExecutive.hh"

int main(int argc, char ** argv)
{
    G4RunManager *runManager = new G4RunManager();

    runManager -> SetUserInitialization(new exampleDetectorConstruction);
    runManager -> SetUserInitialization(new A01PhysicsList);

    runManager -> SetUserAction(new examplePrimaryGeneratorAction);
//  runManager -> SetUserAction(new exampleEventAction);

    runManager -> Initialize();
```

example.cc

- G4VisManager
  - for visualization
- Interactive mode (using prompt)
  - Idle> run BeamOn(1)
- Using macro file
  - $ example run.mac



```
    G4VisManager* visManager = new G4VisExecutive();
    visManager -> Initialize();

    if (argc == 1) {//interactive mode
        G4UIsession* session = new G4UIterminal(new G4UItcsh);
        session -> SessionStart();
        delete session;
    }
    else { // read macro file
        G4String command = "/control/execute ";
        G4String fileName = argv[1];

        G4UImanager* UI = G4UImanager::GetUIpointer();
        UI -> ApplyCommand(command + fileName);

        G4UIsession* session = new G4UIterminal(new G4UItcsh);
        session -> SessionStart();
        delete session;
    }

    delete runManager;
    return 0;
}
```

PhysicsList

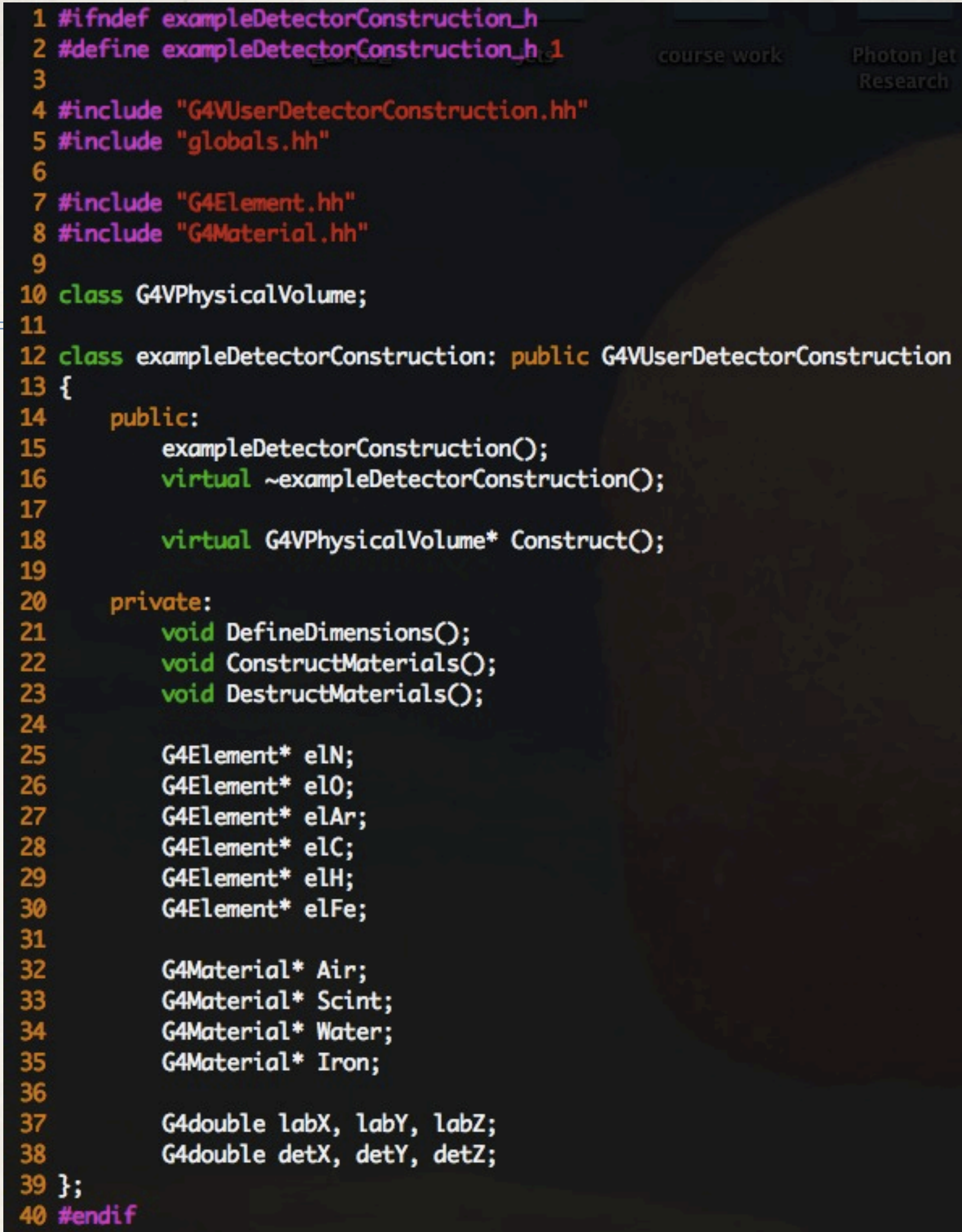
- G4VUserPhysicsList functions
  - ConstructParticle()
  - ConstructProcess()
  - SetCuts()
- Let’s just using well-made source.
  - copy the files to your directories (include and src)

- $G4INSTALL/examples/extended/analysis/A01/
- "include" directory
  - A01PhysicsList.hh
  - A01GeneralPhysics.hh
  - A01MuonPhysics.hh
  - A01EMPhysics.hh
  - A01HadronPhysics.hh
  - A01IonPhysics.hh
- "src" directory
  - A01PhysicsList.cc
  - A01GeneralPhysics.cc
  - A01MuonPhysics.cc
  - A01EMPhysics.cc
  - A01HadronPhysics.cc
  - A01IonPhysics.cc

exampleDetector Construction.hh

- header file
    - like skelecton
    - only definition

- Dimensions
- Materials



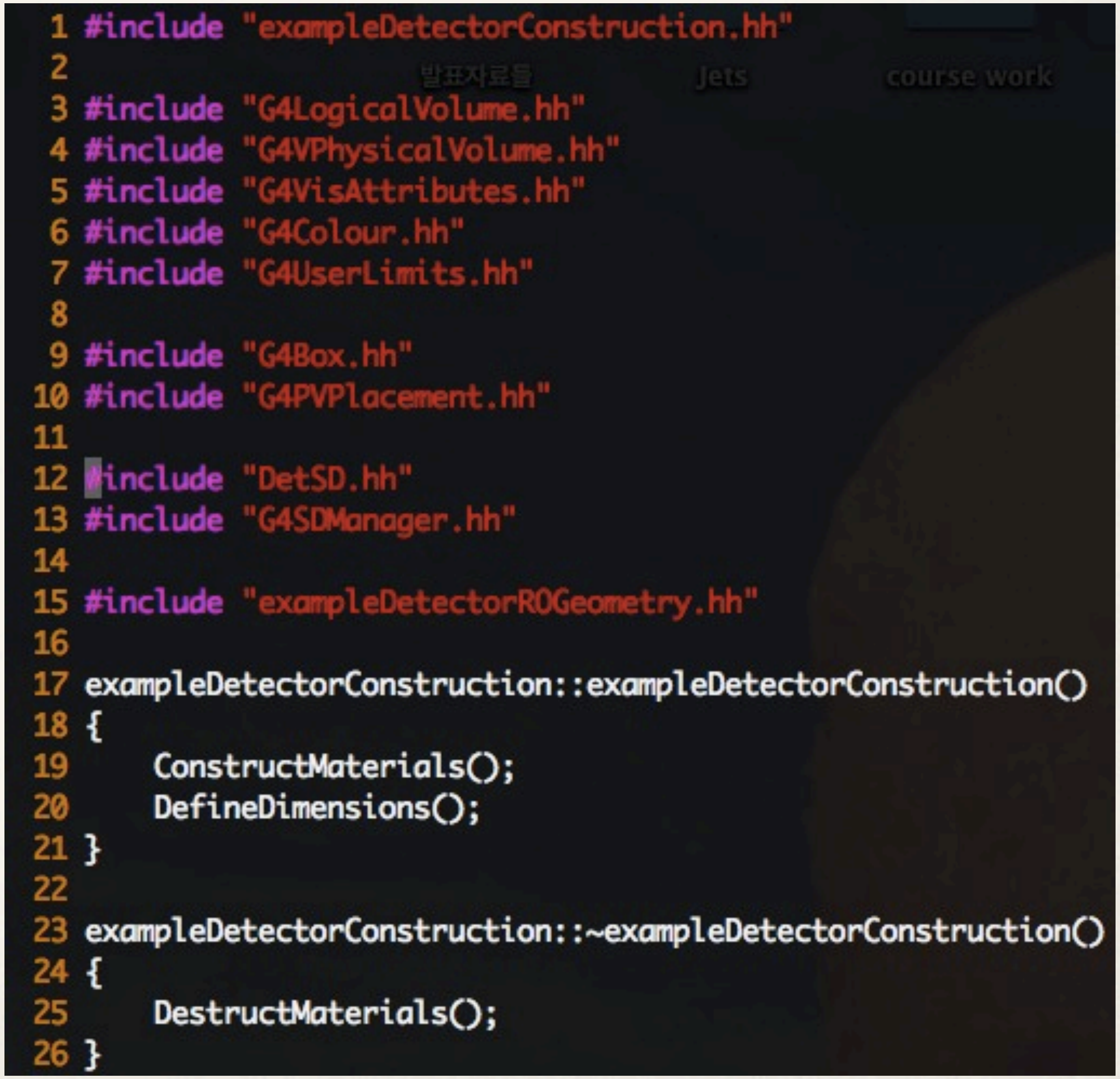
```
#ifndef exampleDetectorConstruction_h
#define exampleDetectorConstruction_h 1

#include "G4VUserDetectorConstruction.hh"
#include "globals.hh"

#include "G4Element.hh"
#include "G4Material.hh"

class G4VPhysicalVolume;

class exampleDetectorConstruction: public G4VUserDetectorConstruction
{
    public:
        exampleDetectorConstruction();
        virtual ~exampleDetectorConstruction();

        virtual G4VPhysicalVolume* Construct();

    private:
        void DefineDimensions();
        void ConstructMaterials();
        void DestructMaterials();

        G4Element* elN;
        G4Element* elO;
        G4Element* elAr;
        G4Element* elC;
        G4Element* elH;
        G4Element* elFe;

        G4Material* Air;
        G4Material* Scint;
        G4Material* Water;
        G4Material* Iron;

        G4double labX, labY, labZ;
        G4double detX, detY, detZ;
};
#endif
```

exampleDetectorConstruction.cc

- source file
  - set up specific code

- #include "header files"

- Constructor
- Destructor

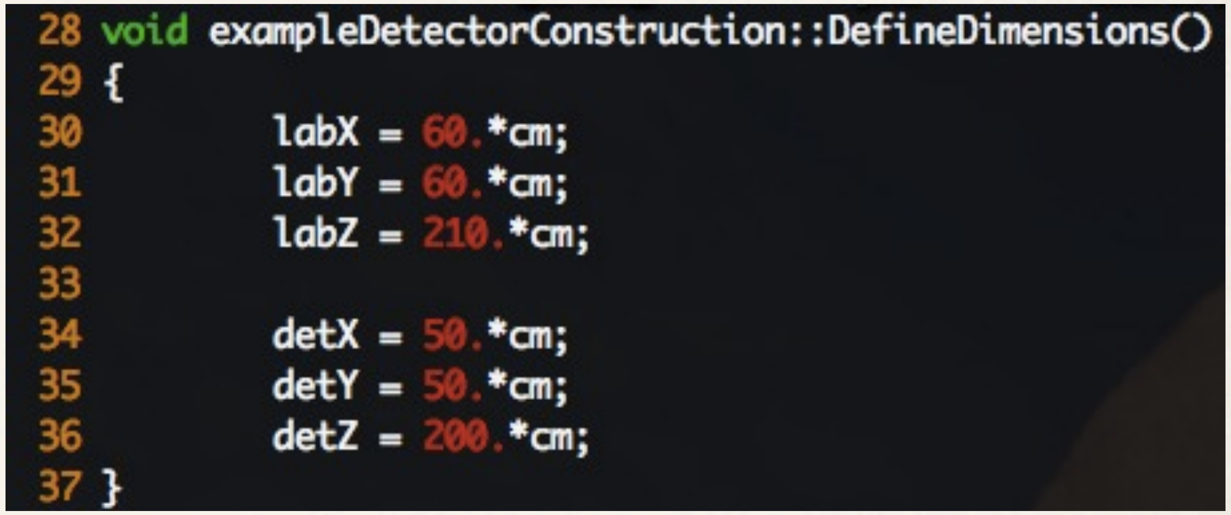


```
#include "exampleDetectorConstruction.hh"

#include "G4LogicalVolume.hh"
#include "G4VPhysicalVolume.hh"
#include "G4VisAttributes.hh"
#include "G4Colour.hh"
#include "G4UserLimits.hh"

#include "G4Box.hh"
#include "G4PVPlacement.hh"

#include "DetSD.hh"
#include "G4SDManager.hh"

#include "exampleDetectorROGeometry.hh"

exampleDetectorConstruction::exampleDetectorConstruction()
{
    ConstructMaterials();
    DefineDimensions();
}

exampleDetectorConstruction::~exampleDetectorConstruction()
{
    DestructMaterials();
}
```

exampleDetectorConstruction.cc

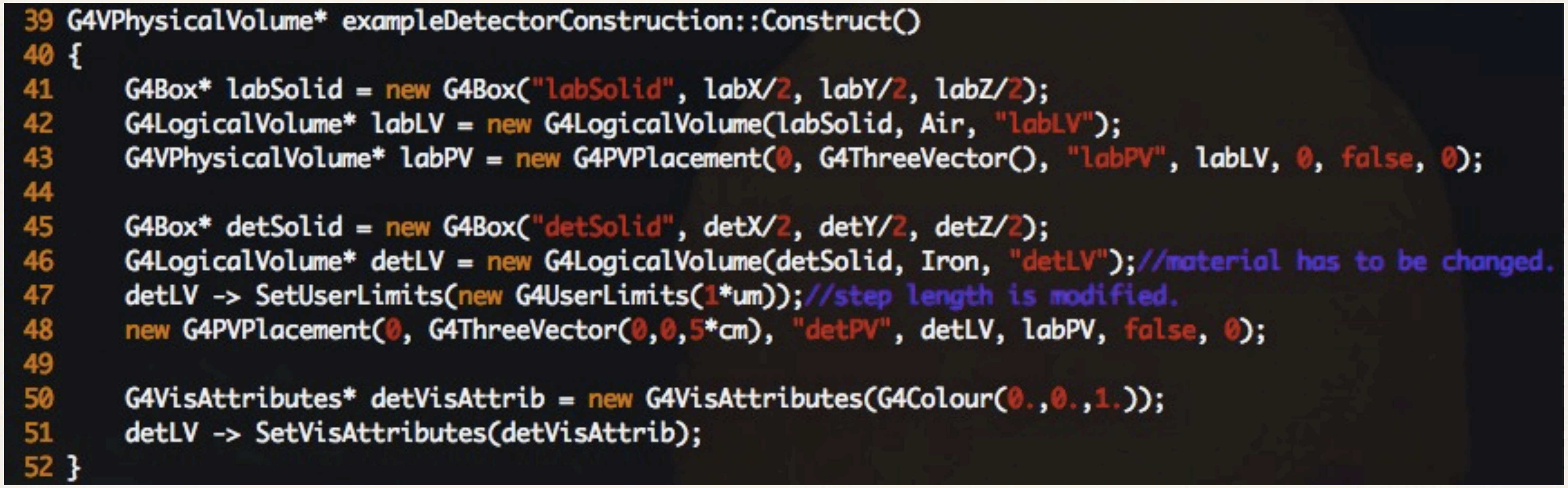
- DefineDimensions()

- we’ve already defined the variables in header file.
    - int labX = ~ (so do not)



```
void exampleDetectorConstruction::DefineDimensions()
{
        labX = 60.*cm;
        labY = 60.*cm;
        labZ = 210.*cm;

        detX = 50.*cm;
        detY = 50.*cm;
        detZ = 200.*cm;
}
```

exampleDetectorConstruction.cc



```
G4VPhysicalVolume* exampleDetectorConstruction::Construct()
{
    G4Box* labSolid = new G4Box("labSolid", labX/2, labY/2, labZ/2);
    G4LogicalVolume* labLV = new G4LogicalVolume(labSolid, Air, "labLV");
    G4VPhysicalVolume* labPV = new G4PVPlacement(0, G4ThreeVector(), "labPV", labLV, 0, false, 0);

    G4Box* detSolid = new G4Box("detSolid", detX/2, detY/2, detZ/2);
    G4LogicalVolume* detLV = new G4LogicalVolume(detSolid, Iron, "detLV");//material has to be changed.
    detLV -> SetUserLimits(new G4UserLimits(1*um));//step length is modified.
    new G4PVPlacement(0, G4ThreeVector(0,0,5*cm), "detPV", detLV, labPV, false, 0);

    G4VisAttributes* detVisAttrib = new G4VisAttributes(G4Colour(0.,0.,1.));
    detLV -> SetVisAttributes(detVisAttrib);
}
```

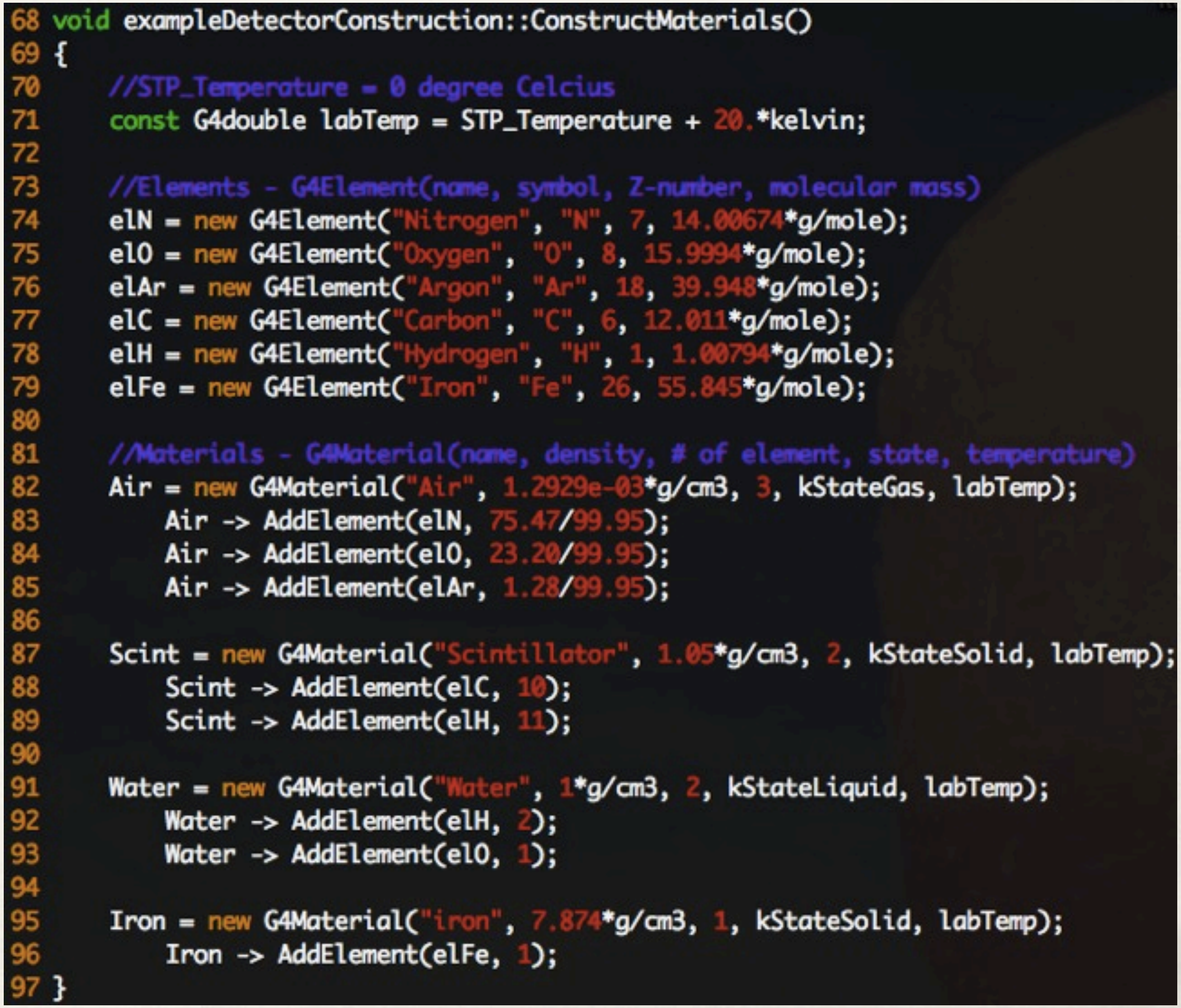
we call 'labPV' as 'world Volume'
the highest volume!

- G4Box(name, x, y, z)
- G4LogicalVolume : geometry + material
- G4PhysicalVolume : LV + position

- G4VisAttributes

exampleDetectorConstruction.cc

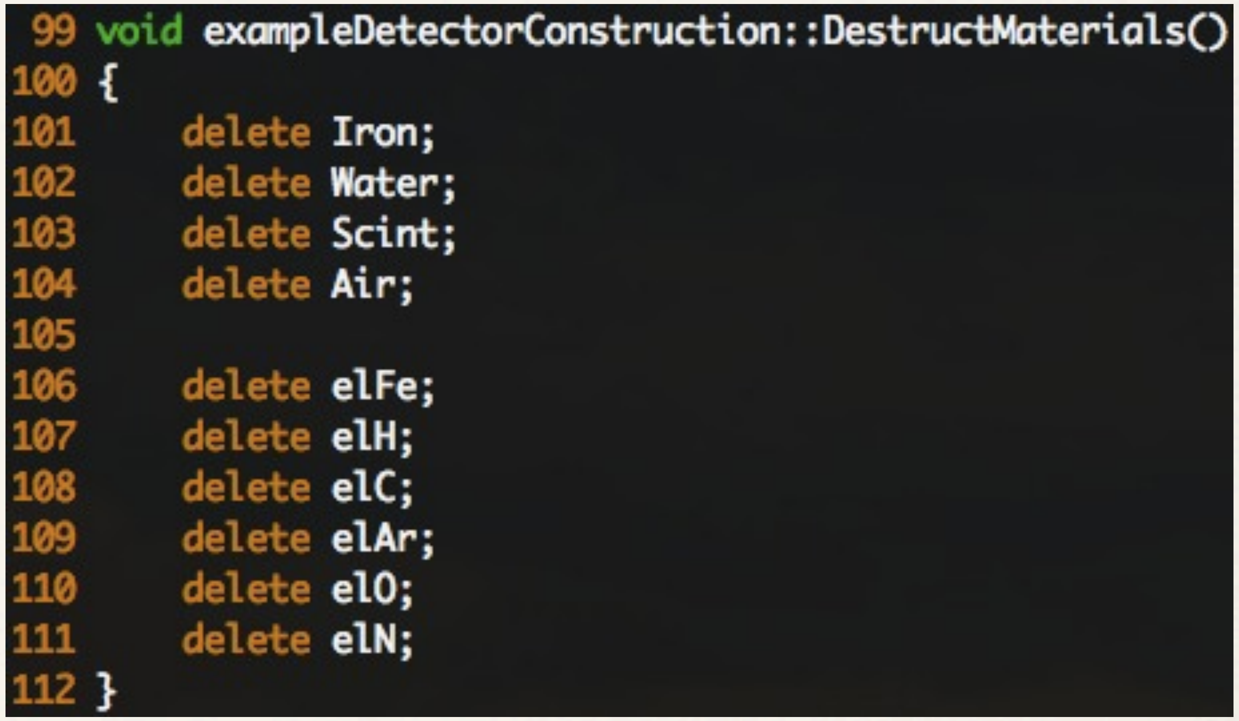
- ConstructMaterials()
    - Temperature
    - G4Element
    - G4Material



```cpp
void exampleDetectorConstruction::ConstructMaterials()
{
    //STP_Temperature = 0 degree Celcius
    const G4double labTemp = STP_Temperature + 20.*kelvin;

    //Elements - G4Element(name, symbol, Z-number, molecular mass)
    elN = new G4Element("Nitrogen", "N", 7, 14.00674*g/mole);
    elO = new G4Element("Oxygen", "O", 8, 15.9994*g/mole);
    elAr = new G4Element("Argon", "Ar", 18, 39.948*g/mole);
    elC = new G4Element("Carbon", "C", 6, 12.011*g/mole);
    elH = new G4Element("Hydrogen", "H", 1, 1.00794*g/mole);
    elFe = new G4Element("Iron", "Fe", 26, 55.845*g/mole);

    //Materials - G4Material(name, density, # of element, state, temperature)
    Air = new G4Material("Air", 1.2929e-03*g/cm3, 3, kStateGas, labTemp);
        Air -> AddElement(elN, 75.47/99.95);
        Air -> AddElement(elO, 23.20/99.95);
        Air -> AddElement(elAr, 1.28/99.95);

    Scint = new G4Material("Scintillator", 1.05*g/cm3, 2, kStateSolid, labTemp);
        Scint -> AddElement(elC, 10);
        Scint -> AddElement(elH, 11);

    Water = new G4Material("Water", 1*g/cm3, 2, kStateLiquid, labTemp);
        Water -> AddElement(elH, 2);
        Water -> AddElement(elO, 1);

    Iron = new G4Material("iron", 7.874*g/cm3, 1, kStateSolid, labTemp);
        Iron -> AddElement(elFe, 1);
}
```

exampleDetectorConstruction.cc

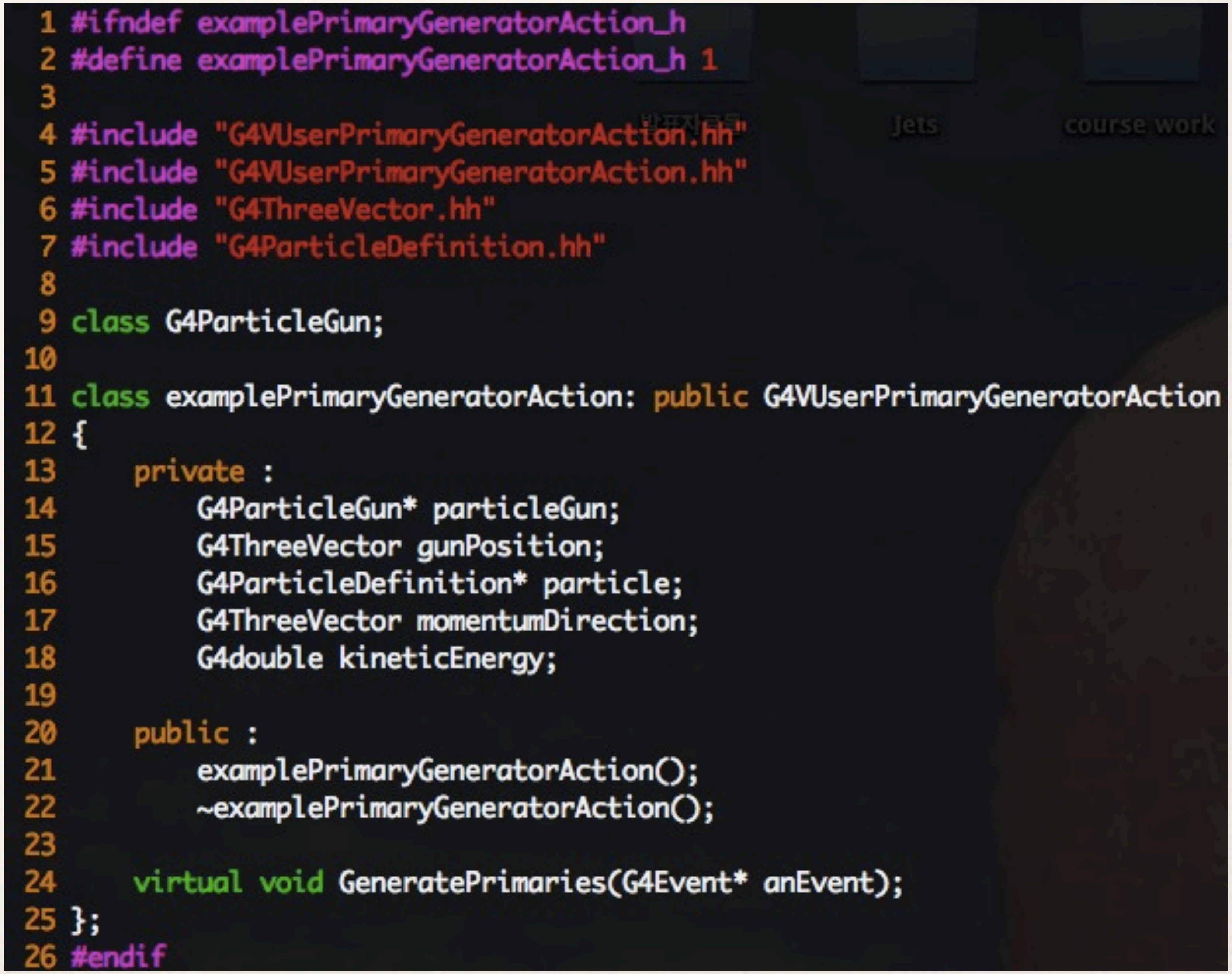
- DestructMaterials()



```cpp
void exampleDetectorConstruction::DestructMaterials()
{
    delete Iron;
    delete Water;
    delete Scint;
    delete Air;

    delete elFe;
    delete elH;
    delete elC;
    delete elAr;
    delete elO;
    delete elN;
}
```

examplePrimaryGeneratorAction.hh

- G4ParticleGun
- below can be initiate here
- gun Position
- particle
- momentum
- kinetic Energy

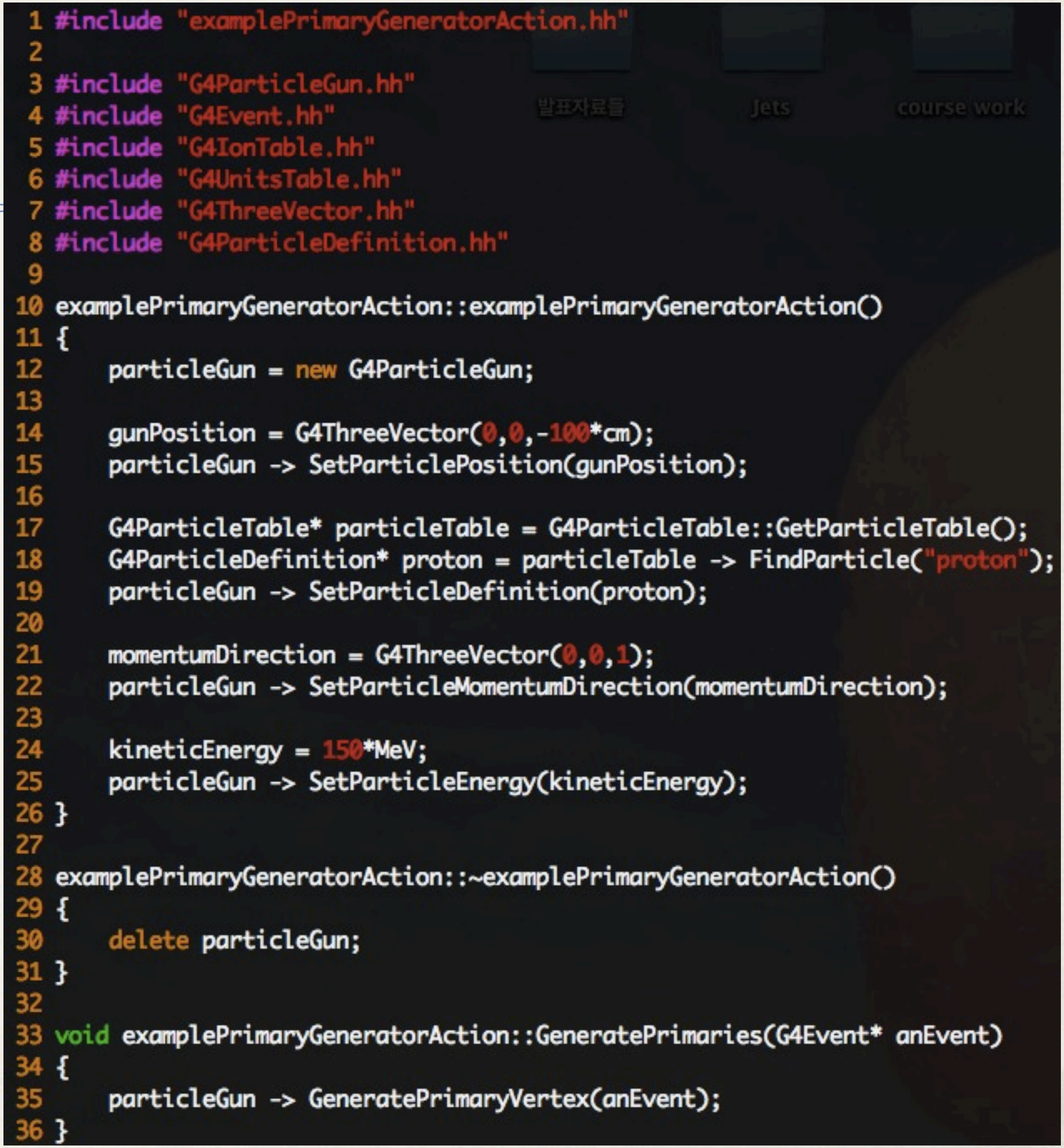


```
#ifndef examplePrimaryGeneratorAction_h
#define examplePrimaryGeneratorAction_h 1

#include "G4VUserPrimaryGeneratorAction.hh"
#include "G4VUserPrimaryGeneratorAction.hh"
#include "G4ThreeVector.hh"
#include "G4ParticleDefinition.hh"

class G4ParticleGun;

class examplePrimaryGeneratorAction: public G4VUserPrimaryGeneratorAction
{
    private :
        G4ParticleGun* particleGun;
        G4ThreeVector gunPosition;
        G4ParticleDefinition* particle;
        G4ThreeVector momentumDirection;
        G4double kineticEnergy;

    public :
        examplePrimaryGeneratorAction();
        ~examplePrimaryGeneratorAction();

    virtual void GeneratePrimaries(G4Event* anEvent);
};
#endif
```

examplePrimaryGeneratorAction.cc

- G4ParticleGun

- below can be initiate here
- gun Position
- particle
- momentum
- kinetic Energy



```
#include "examplePrimaryGeneratorAction.hh"

#include "G4ParticleGun.hh"
#include "G4Event.hh"
#include "G4IonTable.hh"
#include "G4UnitsTable.hh"
#include "G4ThreeVector.hh"
#include "G4ParticleDefinition.hh"

examplePrimaryGeneratorAction::examplePrimaryGeneratorAction()
{
    particleGun = new G4ParticleGun;

    gunPosition = G4ThreeVector(0,0,-100*cm);
    particleGun -> SetParticlePosition(gunPosition);

    G4ParticleTable* particleTable = G4ParticleTable::GetParticleTable();
    G4ParticleDefinition* proton = particleTable -> FindParticle("proton");
    particleGun -> SetParticleDefinition(proton);

    momentumDirection = G4ThreeVector(0,0,1);
    particleGun -> SetParticleMomentumDirection(momentumDirection);

    kineticEnergy = 150*MeV;
    particleGun -> SetParticleEnergy(kineticEnergy);
}

examplePrimaryGeneratorAction::~examplePrimaryGeneratorAction()
{
    delete particleGun;
}

void examplePrimaryGeneratorAction::GeneratePrimaries(G4Event* anEvent)
{
    particleGun -> GeneratePrimaryVertex(anEvent);
}
```

Visualization

- We need Graphic software.
    - OpenGL
    - DAWN

Follow this for what would happen

- $ make
- $ cp bin/Linux-g++/example .
- $ ./example
- Type below
  - Idle >/vis/open OGLIX 900x900
  - Idle >/vis/drawVolume
  - Idle >/vis/viewer/set/viewpointThetaPhi 90 180
  - Idle >/vis/scene/add/axes 0 0 -15 5 cm
  - Idle >/vis/scene/add/trajectories 0
  - Idle >/run/beamOn 2

Follow this for what would happen

- $ ./example vis.mac

- $ dawn g4_00.prim
- $ gv g4_00.eps

Other classes

- **G4Run** : a set of events
  - each beamOn() is each run!
- **G4Event** : a set of 'primaryGeneration'
  - primary particles, primary vertices ... and so on
  - trajectories
  - hit collection (when you have 'Sensitive Detector')
    - 'SD' : an abstract base class which represents a detector for data taking
- **G4Step** : the information of very very small time slice!
- **G4Hit** : snapshot of the physical interaction of a track in the sensitive region of a detector
- **G4Track, G4Trajectory** ...

How to be familiar with Geant4 ?

- ‘Geant4 무작정 따라하기’
- /novice/examples (firstly I recommend N03)

- Googling is very useful! (class reference and lots of examples)

Thank you!