Środowisko programistyczne GEANT4

Leszek Adamczyk

Wydział Fizyki i Informatyki Stosowanej
Akademia Górniczo-Hutnicza

Wykłady w semestrze zimowym 2013/2014

G4VHit - przykład

Przykład z warsztatów implementacji klasy wyprowadzonej z klasy G4VHit

- Wyprowadzimy klasę np. MyPhotonHit zawierającą informację o energii i pozycji każdego fotonu docierającego do sfery otaczającej tarczę berylową.
- Klasa G4VHit zawiera dwie czysto wirtualne metody Draw oraz Print które można zaimplementować jeśli chce się np. wizualizować hity

MyPhotonHit.hh

```
class MyPhotonHit : public G4VHit {

public:
  // Constructor
  MyPhotonHit();

  // Destructor
  virtual ~MyPhotonHit();

  // Methods
  // void Draw()
  // void Print()

  // Photon energy
  inline void SetEnergy(G4double energy) {fEnergy = energy;}
  inline G4double GetEnergy() const {return fEnergy;}

  // Photon position
  inline void SetPosition(G4ThreeVector position) {fPosition = position;}
  inline G4ThreeVector GetPosition() const {return fPosition;}

private:
  //Data Members
 G4double fEnergy;
 G4ThreeVector fPosition;
};
```

MyPhotonHit.cc

```
#include "MyPhotonHit.hh"

MyPhotonHit::MyPhotonHit()
  :fEnergy(0)
  ,fPosition()
{}

MyPhotonHit::~MyPhotonHit() {}
```

G4VSensitiveDetector

Aby zaimplementować własną klasę SensitiveDetector użytkownik musi:

- w konstruktorze dodać nazwy HitCollection's do wektora collectionName oraz dokonać instancji klasy G4VSensitiveDetector
- utworzyć i dopisać HitCollection do obiektu G4HCOfThisEvent w jednej z dwóch metod Initialize() lub EndOfEvent()
- zaimplementować metodę ProcessHits(). W niej tworzymy hity oraz dodajemy je do HitCollection

MyPhotonSD.hh

```
class MyPhotonSD : public G4VSensitiveDetector {

public:

  // Constructor
  MyPhotonSD(const G4String& name);

  // Destructor
  virtual ~MyPhotonSD();

  // Methods
  void Initialize(G4HCofThisEvent* hitsCollectionOfThisEvent);
  G4bool ProcessHits(G4Step* aStep,G4TouchableHistory* history);
  void EndOfEvent(G4HCofThisEvent* hitsCollectionOfThisEvent);

private:

  // Data members
  G4THitsCollection<MyPhotonHit>* fPhotonCollection;
  G4int fPhotonCollectionID;

};
```

MyPhotonSD.cc

```
MyPhotonSD::MyPhotonSD(const G4String& name)
  :G4VSensitiveDetector(name)
{
  collectionName.insert("PhotonCollection");
  fPhotonCollectionID = -1;
}

MyPhotonSD::~MyPhotonSD() {}

void MyPhotonSD::Initialize(G4HCofThisEvent*
                            hitsCollectionOfThisEvent)
{
 // Create a new collection
 fPhotonCollection = new G4THitsCollection<MyPhotonHit>
                  (SensitiveDetectorName, collectionName[0]);
 if( fPhotonCollectionID < 0)
   fPhotonCollectionID = G4SDManager::GetSDMpointer()
                       ->GetCollectionID(fPhotonCollection);
 // Add collection to the events
 hitsCollectionOfThisEvent->AddHitsCollection
                   (fPhotonCollectionID,fPhotonCollection);
}
```

MyPhotonSD.cc

```
G4bool MyPhotonSD::ProcessHits(G4Step* aStep, G4TouchableHistory*)
{
 // Create a new hit
 MyPhotonHit* aHit = new MyPhotonHit();

 // Get energy and position
 G4double energy = aStep->GetTrack()->GetTotalEnergy();
 G4ThreeVector position = aStep->GetTrack()->GetPosition();


 // Set energy and position
 aHit->SetEnergy(energy);
 aHit->SetPosition(position);

 // Insert hit into the collection
 fPhotonCollection->insert(aHit);
}
```

G4Step

Niektóre metody klasy G4Step umożliwiające dostęp do informacji o symulacji:

- GetTrack() - dostęp do informacji o cząstce która generuje krok;
- GetPre(Post)StepPoint() - dostęp do informacji geometrycznych o początku i końcu kroku (pozycja, czas, materiał, ...)
- GetStepLength() - długość kroku;
- GetTotalEnergyDeposit() - całkowity depozyt energii w materiale podczas wykonywania kroku;
- GetDeltaXXXXX() - informacja o zmianie pędu, energii, pozycji,... cząstki generującej krok
- IsFirst(Last)StepInVolume() - wskazuje czy aktualny krok jest pierwszym(ostatnim) w danym obszarze detektora.

G4HCOfThisEvent

- Obiekt klasy G4HCOfThisEvent zawiera wszystkie kolekcje hitów dla danego przypadku.
- Wskaźnik do danej kolekcji może mieć wartość NULL jeśli dana kolekcja nie została utworzona.
- Kolekcje hitów są dostępne za pomocą wskaźników do klasy bazowej G4VHitsCollection, zatem muszą być rzutowane na konkretny typ danej kolekcji.
- Numer identyfikacyjny danej kolekcji jest unikalny i taki sam dla każdego przypadku. Można go uzyskać za pomocą funkcji:
  SDManager::GetCollectionID(nazwaSD/nazwaKolekcji)
  SDManager::GetCollectionID(G4VHitsCollection*)