初期粒子の発生方法

演習パッケージ: P03_PrimaryGenerator

Geant4 10.3.P3準拠

Geant4　HEP/Space/Medicine 講習会資料

大学共同利用機関法人
高エネルギー加速器研究機構

本資料に関する注意

- 本資料の知的所有権は、高エネルギー加速器研究機構およびGeant4 collaborationが有します
- 以下のすべての条件を満たす場合に限り無料で利用することを許諾します
  - 学校、大学、公的研究機関等における教育および非軍事目的の研究開発のための利用であること
    - Geant4の開発者はいかなる軍事関連目的へのGeant4の利用を拒否します
  - このページを含むすべてのページをオリジナルのまま利用すること
    - 一部を抜き出して配布したり利用してはいけません
  - 誤字や間違いと疑われる点があれば報告する義務を負うこと
- 商業的な目的での利用、出版、電子ファイルの公開は許可なく行えません
- 本資料の最新版は以下からダウンロード可能です
  - http://geant4.kek.jp/lecture/
- 本資料に関する問い合わせ先は以下です
  - Email: lecture-feedback@geant4.kek.jp

演習の目標

1. 前演習で使ってきたParticle Gunがどう実装されているかを学ぶ
2. より一般的な粒子発生ツールGeneral Particle Sourceの使い方を学ぶ
3. Particle Gun関数をプログラムコード内で使用すれば、より複雑な初期粒子設定方法が実現できることを学ぶ
4. 発生粒子を磁場中で走らせる方法を学ぶ
5. Geant4の標準的粒子発生ツールを使わず、ユーザが独自に初期粒子を発生させる手法を学ぶ

（これは演習付録として用意されている： 宿題として各自演習）

演習の準備

P03_PrimaryGeneratorプログラムのコピーとファイル構造の確認

課題:0　演習プログラムとして提供されているP03_PrimaryGeneratoの全体をユーザのワークディレクトリにコピーし、そのファイル構造を確認する

[注意]
1. コマンド入力には必ずtcsh補完機能を使う
2. スライドのコマンドを「コピペ」するのは危険

1) 演習プログラム全体を自分のワークデレクトリにコピー

```
$ cd ~/Geant4Tutorial20171129
$ cd UserWorkDir
$ cp -r ../TutorialMaterials/P03_PrimaryGenerator .
```

先ず、演習のルート・ディレクトリに行く

ユーザ作業root dir

P03_PrimaryGeneratorの後ろに ”/”をつけないこと

2) 演習プログラムのファイル構造の確認

```
$ ls P03_PrimaryGenerator
source/    util/
$ ls P03_PrimaryGenerator/source
Application_Main.cc    CMakeLists.txt    include/   src/
```

[注1] mainプログラムは以前の演習(P01_FirstStep_Vis)で使用したものと同じ

mainプログラム [注1]　cmakeビルドファイル　ヘッダファイル　ソースファイル

```
$ ls P03_PrimaryGenerator/source/src
Geometry.cc          PrimaryGenerator.cc      UserActionInitialization.cc
```

[注2] ジオメトリファイルは前の演習で使ったPixel_Oneで使用したものと同じその他のファイルも同じ

ジオメトリ定義ファイル　入射粒子定義ファイル　ユーザ・アクション登録用ファイル

（.ccに対応する.hhはincludeデレクトリにある） [注2]

```
$ ls P03_PrimaryGenerator/util
G4Codes/          Help/          Macros/    Macros_PrimaryGen/
```

本演習で使う様々なGeant4マクロが入っている

演習で使うC++ファイル　救済用スクリプト　アプリ実行用Geant4マクロ

P03_PrimaryGenerator: Gun_One

Particle Gunの実装方法

P03_PrimaryGenerator: Gun_Oneプログラムの概要

■ 演習プログラムの目的

- 今までの演習で使ってきたG4ParticleGunの実装がどのようになされていかを学ぶ

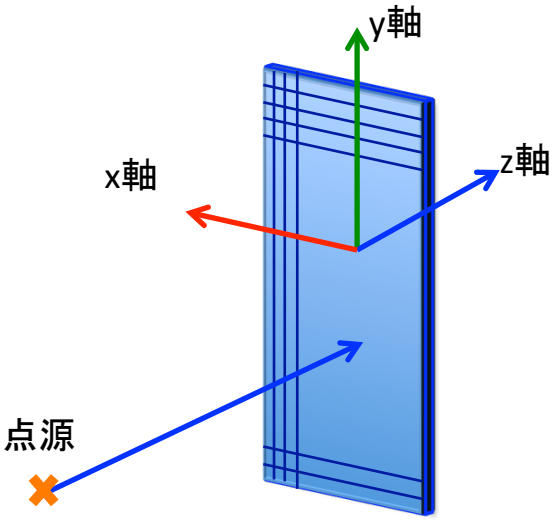
  ［注］この演習では、用意されているプログラムを編集することなく、そのまま使う

■ プログラムの構成

- mainプログラム： P01_FirstStep_Visと同一
- ジオメトリ： P02_Geometryで使ったPixel検出器
- PhysicsList： P01_FirstStep_Visと同一
- PrimaryGenerator: P01_FirstStep_Visと同一

[primaryGeneratorSetup.macの初期設定]

照射粒子：
陽子
50 MeV/c
照射原点：
x = 0,
y = 0,
z = -5.0 cm



P03_PrimaryGenerator: Gun_Oneのビルド

課題：1　P03_PrimaryGenerator: Gun_Oneをビルドして実行ファイルを作成する
(ビルドに必要な全ファイルはコピーしたP03_PrimaryGeneratorにすでに用意されている)

1) P03_PrimaryGeneratorのディレクトリに移行する

```
$ cd ~/Geant4Tutorial20171129/UserWorkDir/P03_PrimaryGenerator
```

← CBDir (Current Base Directory)

2) 前の演習と同じようにbuildデレクトリを作ってビルドを実行　[注1]

```
$ cd ../..
$ mkdir build
$ cd  build
$ cmake ../source
$ make
$ make install
$ cd ..
```

← ビルド作業のデレクトリ名は習慣的にbuildとする
← 作られたbuildに移動
cmakeの慣用句
← CBDirに戻る

[注1]
buildを失敗したら、CBdirのもとで以下のスクリプトを実行すればbuildは自動完了
./util/Help/Build_P03_Gun_One.sh

Application_Main (Gun_One)の実行

課題：2　Application_Main (Gun_One)を実行する

1) プログラム実行を行う作業デレクトリを新たに作成する

```
$ pwd                                                    ← 現デレクトリがCBDirであることを確認：
……/P03_PrimaryGenerator                                   P03_PrimaryGeneratorでなければ、そこへ移動
$ mkdir TestBench                                        ← 作業デレクトリをつくる(名前は自由)
$ cd TestBench
$ cp ../util/Macros/* .                                  ← 用意されているマクロファイルをTestBenchにコピー
$ cp ../util/Macros_PrimaryGen/primaryGeneratorSetup.mac_Gun_One \
     primaryGeneratorSetup.mac                           ← Gun_One用のprimaryGeneratorSetup.macをコピー
overwrite primaryGeneratorSetup.mac? (y/n [n]) y
$ ../bin/Application_Main                                ← TestBenchデレクトリでApplication_Main実行
```

[注]

2) 端末ウインドに以下のメッセージが出力され、続いてQtウインドが開く

```
*************************************************************
 Geant4 version Name: geant4-10-03-patch-03    (20-October-2017)
                      Copyright : Geant4 Collaboration
                      Reference : NIM A 506 (2003), 250-303
                            WWW : http://cern.ch/geant4
*************************************************************

<<< Geant4 Physics List simulation engine: FTFP_BERT 2.0
.....
### Adding tracking cuts for neutron  TimeCut(ns)= 10000  KinEnergyCut(MeV)= 0
Visualization Manager instantiating with verbosity "warnings (3)"...
Visualization Manager initialising...
Registering graphics systems...
.....
```

[注]
前ステップでコピーしたMacros中の
primaryGenerartorSetup.macは
仮のもの。

これをGun_One用に変更する

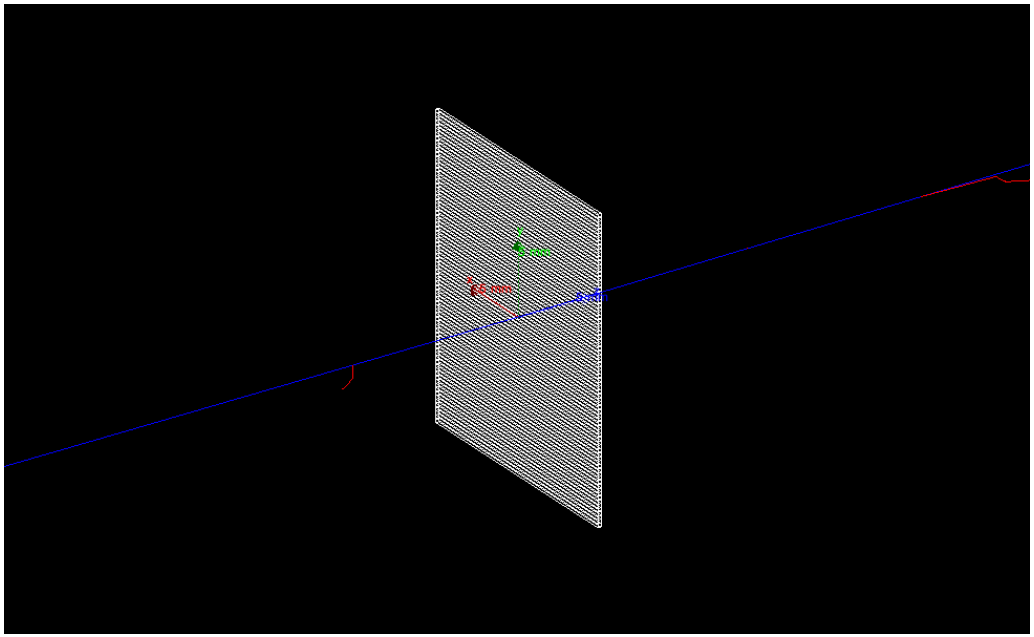
Qtウインドでアプリの動作を確認

課題:3　UI コマンドで事象発生

1) 以下のUIコマンドを先ず入力

```
/run/beamOn 1
```

右図はその出力
（マウスで視点を調整してある）



2) 任意のgunコマンドを実行して何が起こるかを調べる

```
[例]
/run/beamOn 100
```

3) アプリ終了

```
exit
```

P03_PrimaryGenerator: Gun_One　PrimaryGenerator.hhの内容

課題:4　初期粒子定義ヘッダーファイルを理解する

```
$ less ../source/include/PrimaryGenerator.hh
```

```
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// PrimaryGenerator.hh
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
#ifndef PrimaryGenerator_h
#define PrimaryGenerator_h 1

#include "G4VUserPrimaryGeneratorAction.hh"
class G4Event;
class G4ParticleGun;

//-----------------------------------------------------------------------------
  class PrimaryGenerator : public G4VUserPrimaryGeneratorAction
//-----------------------------------------------------------------------------
{
  public:
    PrimaryGenerator();
   ~PrimaryGenerator();

  public:
    void GeneratePrimaries(G4Event*);

  private:
    G4ParticleGun* fpParticleGun;
};
#endif
```

ユーザ定義のクラス名は任意であるが、Hands-onでは、この名前を統一的に使う

ユーザの初期粒子定義は必ずこの仮想クラスを継承して行う

初期粒子定義の実装はこの関数の中で行う

P03_PrimaryGenerator: Gun_One　PrimaryGenerator.ccの内容

課題:5　初期粒子定義実装ファイルを理解する

```
$ less ../source/src/PrimaryGenerator.cc
```

```
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// PrimaryGenerator.cc
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
#include "PrimaryGenerator.hh"
#include "G4ParticleGun.hh"

//--------------------------------------------------------------------------
  PrimaryGenerator::PrimaryGenerator()
  : fpParticleGun(0)
//--------------------------------------------------------------------------
{
  fpParticleGun = new G4ParticleGun();
}

//--------------------------------------------------------------------------
  PrimaryGenerator::~PrimaryGenerator()
//--------------------------------------------------------------------------
{
  delete fpParticleGun;
}

//--------------------------------------------------------------------------
  void PrimaryGenerator::GeneratePrimaries(G4Event* anEvent)
//--------------------------------------------------------------------------
{
  fpParticleGun->GeneratePrimaryVertex(anEvent);
}
```

G4ParticleGunヘッダーファイルのインクルド

ParticleGunオブジェクトの生成

ParticleGunを使って初期粒子を発生

P03_PrimaryGenerator: Gun_One　G4RunManagerへの登録

課題：6　PrimaryGeneratorの登録の構造を理解する

- Primary Generatorの”run manager”への登録は”UserActionInitialization”オブジェクト経由で自動的に行われる　← Primary Generator登録はUser Actionsの一つであ
- Primary Generator登録の流れ

  G4PrimaryGenerator (Primarygenerator) → G4VUserActionInitialization (UserActionInitialization) → G4RunManager

[注]　演習の全てを通して’UserActionInitialization’クラスは’PrimaryGenerator’オブジェクトを登録するだけのために存在している

UserActionInitialization.cc

```
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// UserActionInitialization.cc
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
#include "UserActionInitialization.hh"
#include "PrimaryGenerator.hh"

//---------------------------------------------------------------
  UserActionInitialization::UserActionInitialization()
  : G4VUserActionInitialization()
//---------------------------------------------------------------
{}

//---------------------------------------------------------------
  UserActionInitialization::~UserActionInitialization()
//---------------------------------------------------------------
{}

//---------------------------------------------------------------
  void UserActionInitialization::Build() const
//---------------------------------------------------------------
{
  SetUserAction( new PrimaryGenerator() );
}
```

Application_Main.cc

```
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// Geant4 Application: Tutorial course for Hep/Medicine Users
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
#include "Geometry.hh"
#include "UserActionInitialization.hh"

#include "G4RunManager.hh"
#include "G4UIExecutive.hh"
#include "FTFP_BERT.hh"

//---------------------------------------------------------------
  int main( int argc, char** argv )
//---------------------------------------------------------------
{
// Construct the default run manager
   G4RunManager* runManager = new G4RunManager;

// Set up mandatory user initialization: Geometry
   runManager->SetUserInitialization( new Geometry );

// Set up mandatory user initialization: Physics-List
   runManager->SetUserInitialization( new FTFP_BERT );

// Set up user initialization: User Actions
   runManager->SetUserInitialization( new UserActionInitialization );

// Initialize G4 kernel
   runManager->Initialize();

// Start interactive session
   G4UIExecutive* uiExec = new G4UIExecutive(argc, argv, "tcsh");
   uiExec->SessionStart();

// Job termination
   delete uiExec;
   delete runManager;

   return 0;
}
```

PrimaryGeneratorの UserActionInitialization への登録

UserActionInitialization のRunManagerへの登録

KEK 大学共同利用機関法人 高エネルギー加速器研究機構　Geant 4

P03_PrimaryGenerator: GPS

General Particle Sourceの使い方

P03_PrimaryGenerator: GPSプログラムの概要

■ 演習プログラムの目的

- 今までの演習ではPrimary GeneratorとしてG4ParticleGunを使用。本演習ではツールキットが提供するもう一つのGeneratorであるG4GeneralParticleSource（GPS）のセットアップ法を学ぶ

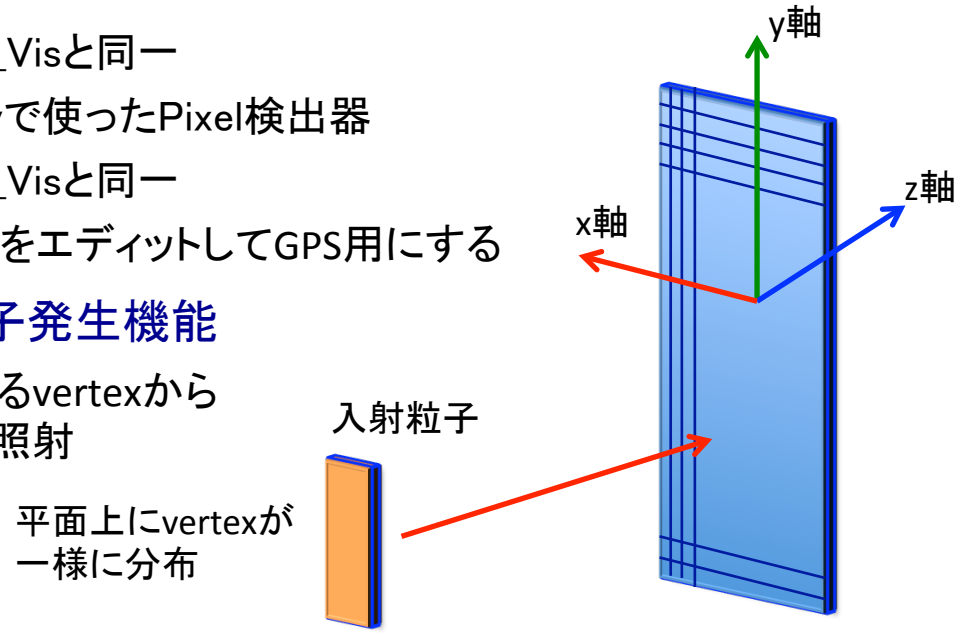
  ［注］この演習では、受講者はソースコードを書き、それを動かしてみる
- 新たに作るプログラムはP03_PrimaryGenerator：GPSとよぶ
- GPSの基本的なUIコマンドの使用方法を学ぶ

■ プログラムの構成

- mainプログラム：　P01_FirstStep_Visと同一
- ジオメトリ：　P02_Geometryで使ったPixel検出器
- PhysicsList：　P01_FirstStep_Visと同一
- PrimaryGenerator：　前演習のものをエディットしてGPS用にする

■ 組み込まれているジオメトリと粒子発生機能

- 粒子発生：　平面に一様に分布するvertexから粒子をPixelに向けて照射



P03_PrimaryGenerator: GPSの実装　(.hh ファイル)

課題：1　現在のPrimaryGenerator.hhからの変更箇所を理解する
［注］ 以下のソースコードは変更が完了したものを示す

エディット手順は
次のスライド参照

```
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// PrimaryGenerator.hh
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
#ifndef PrimaryGenerator_h
#define PrimaryGenerator_h 1

#include "G4VUserPrimaryGeneratorAction.hh"
class G4Event;
class G4GeneralParticleSource;

//--------------------------------------------------------------------------
  class PrimaryGenerator : public G4VUserPrimaryGeneratorAction
//--------------------------------------------------------------------------
{
  public:
    PrimaryGenerator();
   ~PrimaryGenerator();

  public:
    void GeneratePrimaries(G4Event*);

  private:
    G4GeneralParticleSource* fpParticleGPS;
};
#endif
```

変更

forward declarationを
GPSにする

変更

PrimaryGeneratorへの
pointer typeをGPS

P03_PrimaryGenerator: GPSの実装 (.hh ファイル) – つづき

課題:2 現在のPrimaryGenerator.hhをエディットしてGPSを実装する

.hhのあるdir

1) 現在のPrimaryGenerator.hhとGPS用(模範解答)の相違をcolordiffで端末に表示する

```
$ cd ~/Geant4Tutorial20171129/UserWorkDir/P03_PrirmaryGenerator/source/include
$ colordiff -y --width=200  PrimaryGenerator.hh \
                            ../../util/G4Codes/PrimaryGenerator.hh_GPS
```

Gun_Oneファイル

```
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// PrimaryGenerator.hh
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
#ifndef PrimaryGenerator_h
#define PrimaryGenerator_h 1

#include "G4VUserPrimaryGeneratorAction.hh"
class G4Event;
class G4ParticleGun;

//-----------------------------------------------------------------------------
  class PrimaryGenerator : public G4VUserPrimaryGeneratorAction
//-----------------------------------------------------------------------------
{
  public:
    PrimaryGenerator();
   ~PrimaryGenerator();

  public:
    void GeneratePrimaries(G4Event*);

  private:
    G4ParticleGun* fpParticleGun;
};
#endif
```

変更ライン印

GPSファイル

```
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// PrimaryGenerator.hh
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
#ifndef PrimaryGenerator_h
#define PrimaryGenerator_h 1

#include "G4VUserPrimaryGeneratorAction.hh"
class G4Event;
class G4GeneralParticleSource;

//-----------------------------------------------------------------------------
  class PrimaryGenerator : public G4VUserPrimaryGeneratorAction
//-----------------------------------------------------------------------------
{
  public:
    PrimaryGenerator();
   ~PrimaryGenerator();

  public:
    void GeneratePrimaries(G4Event*);

  private:
    G4GeneralParticleSource* fpParticleGPS;
};
#endif
```

2) 現在のPrimaryGenerator.ccをエディターで開き、colordiffの結果を参照しながらGPSを実装する

```
$ pwd
$ gedit PrimaryGenerator.hh
```

PrimaryGenerator.hhのあるdirであることを確認 – 異なれば移動

[注1] 変更をタイプするのが大変なら、以下のファイルをターミナルで表示して、変更箇所をコピペすること
$ less ../../util/G4Codes/PrimaryGenerator.hh_GPS
[注2] コピペがうまくできない場合、以下のコマンドを実行して模範解答を全コピーする
$ cp ../../util/G4Codes/PrimaryGenerator.hh_GPS PrimaryGenerator.hh

P03_PrimaryGenerator: GPSの実装　(.cc ファイル)

課題：3　現在のPrimaryGenerator.ccからの変更箇所を理解する
［注］ 以下のソースコードは変更が完了したものを示す

エディット手順は
次のスライド参照

```
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// PrimaryGenerator.cc
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

#include "PrimaryGenerator.hh"
#include "G4GeneralParticleSource.hh"
#include "G4ParticleTable.hh"
#include "G4ParticleDefinition.hh"
#include "globals.hh"

//--------------------------------------------------------------------------
  PrimaryGenerator::PrimaryGenerator()
: fpParticleGPS(0)
//--------------------------------------------------------------------------
{
fpParticleGPS = new G4GeneralParticleSource();
}

//--------------------------------------------------------------------------
  PrimaryGenerator::~PrimaryGenerator()
//--------------------------------------------------------------------------
{
delete fpParticleGPS;
}

//--------------------------------------------------------------------------
  void PrimaryGenerator::GeneratePrimaries(G4Event* anEvent)
//--------------------------------------------------------------------------
{
fpParticleGPS->GeneratePrimaryVertex(anEvent);
}
```

追加

#include "G4ParticleGun.hh"
の行がこれらに置き換わっている

変更

変更

GPSオブジェクトの生成

変更

GPSオブジェクトの消去

変更

GPSで一つのeventを生成

P03_PrimaryGenerator: GPSの実装 (.cc ファイル) – つづき

課題:4 現在のPrimaryGenerator.ccをエディットしてGPSを実装する

.ccのあるdir

1) 現在のPrimaryGenerator.ccとGPS用(模範解答)の相違をcolordiffで端末に表示する

```
$ cd ~/Geant4Tutorial20171129/UserWorkDir/P03_PrirmaryGenerator/source/src
$ colordiff -y --width=200  PrimaryGenerator.cc \
                            ../../util/G4Codes/PrimaryGenerator.cc_GPS
```

Gun_Oneファイル

```
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// PrimaryGenerator.cc
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

#include "PrimaryGenerator.hh"
#include "G4ParticleGun.hh"



//---------------------------------------------------------------------------
  PrimaryGenerator::PrimaryGenerator()
  : fpParticleGun(0)
//---------------------------------------------------------------------------
{
  fpParticleGun = new G4ParticleGun();
}

//---------------------------------------------------------------------------
  PrimaryGenerator::~PrimaryGenerator()
//---------------------------------------------------------------------------
{
  delete fpParticleGun;
}

//---------------------------------------------------------------------------
  void PrimaryGenerator::GeneratePrimaries(G4Event* anEvent)
//---------------------------------------------------------------------------
{
  fpParticleGun->GeneratePrimaryVertex(anEvent);
}
```

GPSファイル

```
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// PrimaryGenerator.cc
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
>
#include "PrimaryGenerator.hh"
|  #include "G4GeneralParticleSource.hh"
>  #include "G4ParticleTable.hh"
>  #include "G4ParticleDefinition.hh"
>  #include "globals.hh"

//---------------------------------------------------------------------------
  PrimaryGenerator::PrimaryGenerator()
|   : fpParticleGPS(0)
//---------------------------------------------------------------------------
{
|   fpParticleGPS = new G4GeneralParticleSource();
}

//---------------------------------------------------------------------------
  PrimaryGenerator::~PrimaryGenerator()
//---------------------------------------------------------------------------
{
|   delete fpParticleGPS;
}

//---------------------------------------------------------------------------
  void PrimaryGenerator::GeneratePrimaries(G4Event* anEvent)
//---------------------------------------------------------------------------
{
|   fpParticleGPS->GeneratePrimaryVertex(anEvent);
}
```

2) 現在のPrimaryGenerator.ccをエディターで開き、colordiffの結果を参照しながらGPSを実装する

```
$ pwd
$ gedit PrimaryGenerator.cc&
```

PrimaryGenerator.ccのあるdirであることを確認 – 異なれば移動

[注1] 変更をタイプするのが大変なら、以下のファイルをターミナルで表示して、変更箇所をコピペすること
$ less ../../util/G4Codes/PrimaryGenerator.cc_GPS
[注2] コピペがうまくできない場合、以下のコマンドを実行して模範解答を全コピーする
$ cp ../../util/G4Codes/PrimaryGenerator.cc_GPS PrimaryGenerator.cc

P03_PrimaryGenerator: GPSアプリケーションのビルド

課題:５　新たに作ったPrimaryGenerator.hh/.ccを用いてアプリケーションをビルドする

1） 現在のベースデレクトリに移動

```
$ cd ~/Geant4Tutorial20171129/UserWorkDir/P03_PrimaryGenerator
```

← CBDir (Current Base Directory)

2) buildデレクトリでビルドを実行　[注1]

```
$ \rm -r bin
$ cd  build
$ \rm -r *
$ cmake ../source
$ make
$ make install
$ cd ..
```

- \rm -r bin ← 前のプログラムをビルドした時に作らたbinは消去(省略可)
- cd build ← 前のプログラムをビルドした時に作られたbuildに移動
- \rm -r * ← 前のビルドで作られているファイル類を全て消去
- cmake ../source / make / make install ← cmakeの慣用句
- cd .. ← CBDirに戻る

[注1]
buildのステップでerrorが出て、どうしても
アプリケーションが作れない場合、CBdirのもとで
以下のスクリプトを実行すれば、模範解答の
PrimaryGenerator:GPSをもとにbuildを
自動完了することができる

./util/Help/Build_P03_GPS.sh

Application_Main (GPS)の実行

課題：6　Application_Main (GPS)を実行する

1) プログラム実行を行う作業デレクトリに移動

```
$ pwd                                          ← 現デレクトリがCBDirであることを確認：
……/P03_PrimaryGenerator                         P03_PrimaryGeneratorでなければ、そこへ移動
$ cd TestBench                                 ← 作業デレクトリに移動
$ cp ../util/Macros_PrimaryGen/primaryGeneratorSetup.mac_GPS \
     primaryGeneratorSetup.mac                 ← GPS用のprimaryGeneratorSetup.macをコピー
overwrite primaryGeneratorSetup.mac? (y/n [n]) y
$ ../bin/Application_Main                      ← TestBenchデレクトリでApplication_Main実行
```

2) 端末ウインドに以下のメッセージが出力され、続いてQtウインドが開く

```
**************************************************************
 Geant4 version Name: geant4-10-03-patch-03    (20-October-2017)
                      Copyright : Geant4 Collaboration
                      Reference : NIM A 506 (2003), 250-303
                            WWW : http://cern.ch/geant4
**************************************************************

<<< Geant4 Physics List simulation engine: FTFP_BERT 2.0
.....
### Adding tracking cuts for neutron  TimeCut(ns)= 10000  KinEnergyCut(MeV)= 0
Visualization Manager instantiating with verbosity "warnings (3)"...
Visualization Manager initialising...
Registering graphics systems...
.....
```

Qtウインドでアプリの動作を確認

課題：7　GPS UI コマンドを使って
動作をチェックする

1) 以下のUIコマンドを先ず入力

```
/run/beamOn 50
```

右図はその出力
（マウスで視点を移動させ見た目を調整している）

- primaryGeneratorSetup.macの内容

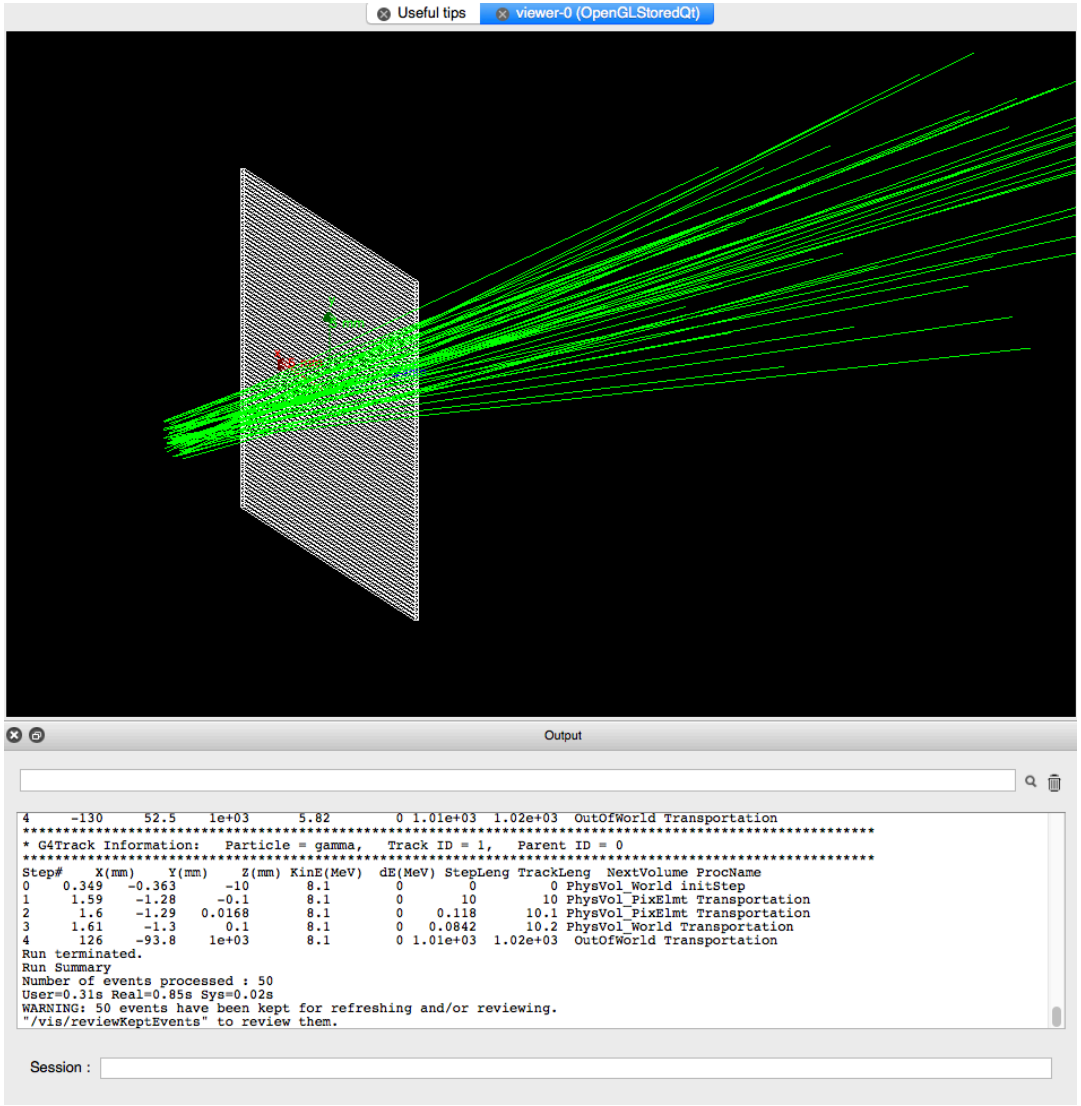
```
/gps/particle gamma
/gps/pos/type Plane
/gps/pos/shape Square
/gps/pos/centre 0 0 -1 cm
/gps/pos/halfx 1 mm
/gps/pos/halfy 1 mm/gps/ang/type cos
/gps/ang/maxtheta 10 deg
/gps/ene/type Lin
/gps/ene/min 2 MeV
/gps/ene/max 10 MeV
/gps/ene/gradient 1
/gps/ene/intercept 1
```

2) 以下の/gpsコマンドで粒子を変更する

```
/gps/particle proton
/run/beamOn 50
```

3) アプリ終了

```
exit
```



P03_PrimaryGenerator: Gun_Two

Particle Gun提供関数を使った初期粒子設定

P03_PrimaryGenerator: Gun_Twoプログラムの概要

■ 演習プログラムの目的

- 最初の演習ではG4ParticleGunをコマンド・レベルで使用した。この演習では初期粒子定義をG4ParticleGunが提供している関数を使いプログラム・レベルで行う手法を学ぶ。これにより標準のParticle Gunコマンドでは実現できない粒子発生が可能となる

  ［注1］ 直前の演習で学んだGPSはコマンドレベルでしか使用できない

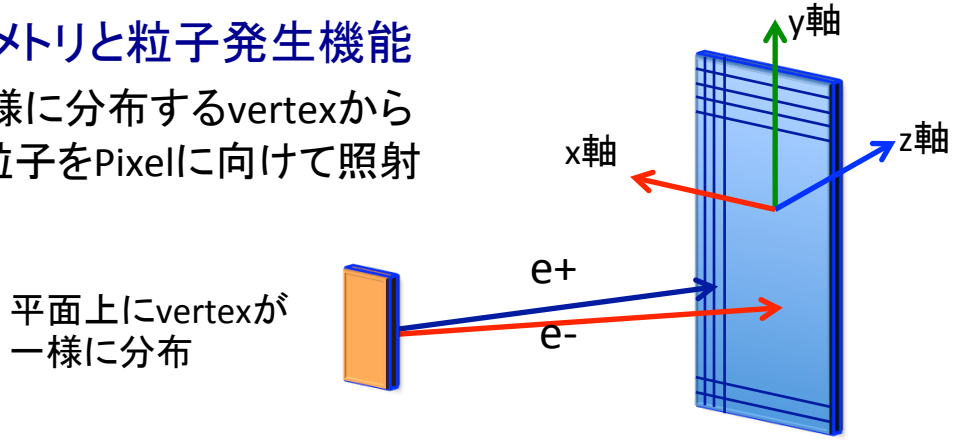
  ［注2］ この演習では、用意されているプログラムを編集することなく、そのまま使う

■ プログラムの構成

- mainプログラム： P01_FirstStep_Visと同一
- ジオメトリ： P02_Geometryで使ったPixel検出器
- PhysicsList： P01_FirstStep_Visと同一
- PrimaryGenerator: 前演習のコードを提供されているものに置き換える

■ 組み込まれているジオメトリと粒子発生機能

- 粒子発生： 平面に一様に分布するvertexから
  e+とe-の粒子をPixelに向けて照射



P03_PrimaryGenerator: Gun_Twoのビルド

課題：1　P03_PrimaryGenerator: Gun_Twoをビルドして実行ファイルを作成する

1）ソースファイルPrimaryGenerator.cc_Gun_TwoをPrimaryGenerator.ccにコピー

```
$ cd ~/Geant4Tutorial20171129/UserWorkDir/P03_PrimaryGenerator
$ cd source/src
$ cp ../../util/G4Codes/PrimaryGenerator.cc_Gun_Two PrimaryGenerator.cc
overwrite PrimaryGenerator.cc? (y/n [n]) y
```

CBDir (Current Base Directory)

2）ヘッダファイルPrimaryGenerator.cc_Gun_TwoをPrimaryGenerator.hhにコピー

```
$ cd ../include
$ cp ../../util/G4Codes/PrimaryGenerator.hh_Gun_Two PrimaryGenerator.hh
overwrite PrimaryGenerator.hh? (y/n [n]) y
```

3）buildデレクトリでビルドを実行　[注1]

```
$ cd ../..
$ \rm -r bin
$ cd  build
$ \rm -r *
$ cmake ../source
$ make
$ make install
$ cd ..
```

CBDir (Current Base Directory)へ戻る
前のプログラムをビルドした時に作らたbinは消去(省略可)
前のプログラムをビルドした時に作られたbuildに移動
前のビルドで作られているファイル類を全て消去

cmakeの慣用句

[注1]
buildを失敗したら、CBdirのもとで以下のスクリプトを実行すればbuildは自動完了
./util/Help/Build_P03_Gun_Two.sh

Application_Main (Gun_Two)の実行

課題：2　Application_Main (Gun_Two)を実行する

1) プログラム実行を行う作業デレクトリに移動

```
$ pwd                                                    ← 現デレクトリがCBDirであることを確認：
……/P03_PrimaryGenerator                                   P03_PrimaryGeneratorでなければ、そこへ移動
$ cd TestBench                                           ← 作業デレクトリに移動
$ cp ../util/Macros_PrimaryGen/primaryGeneratorSetup.mac_Gun_Two \
     primaryGeneratorSetup.mac                           ← Gun_Two用のprimaryGeneratorSetup.macをコピー
overwrite primaryGeneratorSetup.mac? (y/n [n]) y
$ ../bin/Application_Main                                ← TestBenchデレクトリでApplication_Main実行
```

2) 端末ウインドに以下のメッセージが出力され、続いてQtウインドが開く

```
**************************************************************
 Geant4 version Name: geant4-10-03-patch-03    (20-October-2017)
                      Copyright : Geant4 Collaboration
                      Reference : NIM A 506 (2003), 250-303
                            WWW : http://cern.ch/geant4
**************************************************************

<<< Geant4 Physics List simulation engine: FTFP_BERT 2.0
.....
### Adding tracking cuts for neutron  TimeCut(ns)= 10000  KinEnergyCut(MeV)= 0
Visualization Manager instantiating with verbosity "warnings (3)"...
Visualization Manager initialising...
Registering graphics systems...
.....
```

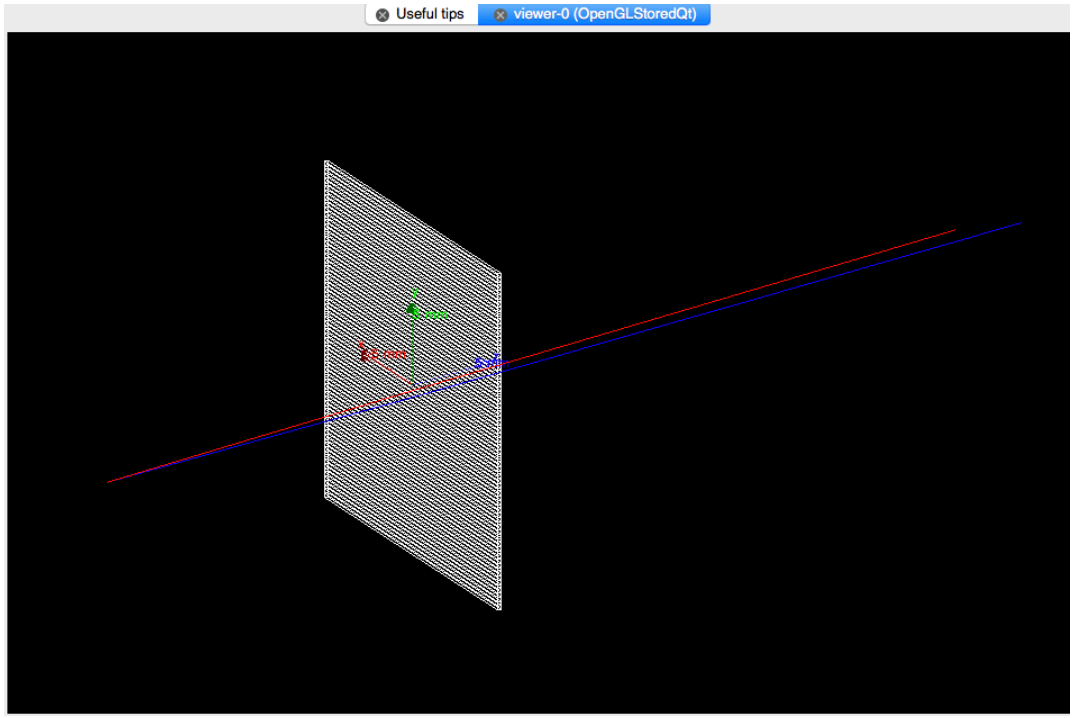
Qtウインドでアプリの動作を確認

課題:3　UI コマンドで事象発生

1) 以下のUIコマンドを先ず入力

```
/run/beamOn 1
```

右図はその出力
（マウスで視点を調整してある）



2) 照射回数を自由に変更してみる

3) アプリ終了

```
exit
```

P03_PrimaryGenerator: Gun_Two　PrimaryGenerator.hhの内容

課題:4　初期粒子定義ヘッダーファイルを理解する

```
$ less ../source/include/PrimaryGenerator.hh
```

```
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// PrimaryGenerator.hh
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
#ifndef PrimaryGenerator_h
#define PrimaryGenerator_h 1

#include "G4VUserPrimaryGeneratorAction.hh"
class G4Event;
class G4ParticleGun;

//------------------------------------------------------------------------------
  class PrimaryGenerator : public G4VUserPrimaryGeneratorAction
//------------------------------------------------------------------------------
{
  public:
    PrimaryGenerator();
   ~PrimaryGenerator();

  public:
    void GeneratePrimaries(G4Event*);

  private:
    G4ParticleGun* fpParticleGun_1;
    G4ParticleGun* fpParticleGun_2;
};
#endif
```

Particle Gunを使う

初期粒子定義の実装はこの関数の中で行う

Particle Gunを2個作るのでポインターを2個用意する

P03_PrimaryGenerator: Gun_Two　PrimaryGenerator.ccの内容

課題:5　初期粒子定義実装ファイルを理解する

```
$ less ../source/src/PrimaryGenerator.cc
```

```
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// PrimaryGenerator.cc
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
#include "PrimaryGenerator.hh"
#include "G4ParticleTable.hh"
#include "G4ParticleGun.hh"
#include "G4SystemOfUnits.hh"
#include "Randomize.hh"

//---------------------------------------------------------------------------
  PrimaryGenerator::PrimaryGenerator()
  : fpParticleGun_1(0), fpParticleGun_2(0)
//---------------------------------------------------------------------------
{
// Particle table
   G4ParticleTable* particleTable = G4ParticleTable::GetParticleTable();

// 1st gun - setup for fixed parameters   一番目のGunの設定
   fpParticleGun_1  = new G4ParticleGun();
   {
     G4String particleName = "e-";
     G4double momentum = 50.0*MeV;
     G4double angle = 1.0*deg;
     G4ThreeVector momentumDirection = G4ThreeVector(0.0, 0.0, 1.0).rotateY(angle);
     G4ParticleDefinition* particle = particleTable->FindParticle(particleName);

     fpParticleGun_1->SetParticleDefinition(particle);
     fpParticleGun_1->SetParticleMomentum(momentum);
     fpParticleGun_1->SetParticleMomentumDirection(momentumDirection);
   }

// 2nd gun - setup for fixed parameters   一番目のGunの設定
   fpParticleGun_2  = new G4ParticleGun();
   {
     G4String particleName = "e+";
     G4double momentum = 50.0*MeV;
     G4double angle = -1.0*deg;
     G4ThreeVector momentumDirection = G4ThreeVector(0.0, 0.0, 1.0).rotateY(angle);
     G4ParticleDefinition* particle = particleTable->FindParticle(particleName);

     fpParticleGun_2->SetParticleDefinition(particle);
     fpParticleGun_2->SetParticleMomentum(momentum);
     fpParticleGun_2->SetParticleMomentumDirection(momentumDirection);
   }
}
```

一番目のGunを作る

一番目のGunの粒子発射条件の設定

二番目のGunを作る

二番目のGunの粒子発射条件の設定

コードは次のスライドにつづく

P03_PrimaryGenerator: Gun_Two　PrimaryGenerator.hhの内容(つづき)

課題：5　初期粒子定義実装ファイルを理解する （前のスライドからのつづき）

```
//....oooOOOooo.....oooOOOooo.....oooOOOooo.....oooOOOooo.....oooOOOooo....
  PrimaryGenerator::~PrimaryGenerator()
//....oooOOOooo.....oooOOOooo.....oooOOOooo.....oooOOOooo.....oooOOOooo....
{
  delete fpParticleGun_1;
  delete fpParticleGun_2;
}

//....oooOOOooo.....oooOOOooo.....oooOOOooo.....oooOOOooo.....oooOOOooo....
  void PrimaryGenerator::GeneratePrimaries(G4Event* anEvent)
//....oooOOOooo.....oooOOOooo.....oooOOOooo.....oooOOOooo.....oooOOOooo....
{
// Gun position - randomization
   G4double pos_X = 2.0*mm*(G4UniformRand()-0.5);
   G4double pos_Y = 2.0*mm*(G4UniformRand()-0.5);
   G4double pos_Z = -2.0*cm;
   G4ThreeVector position = G4ThreeVector(pos_X, pos_Y, pos_Z);

   fpParticleGun_1->SetParticlePosition(position);
   fpParticleGun_2->SetParticlePosition(position);

// Generate primaries
   fpParticleGun_1->GeneratePrimaryVertex(anEvent);
   fpParticleGun_2->GeneratePrimaryVertex(anEvent);
}
```

Gunの位置(vertexの位置)をevent発生ごとに設定

vertex位置を乱数で決定

二つのGunに上のvertex位置を設定

event発生（初期粒子発生）

二つのGunを発射

P03_PrimaryGenerator: MagField

二粒子発生Primary Generatorでジオメトリに磁場を入れる

P03_PrimaryGenerator: MagFieldプログラムの概要

■ 演習プログラムの目的

- 直前に学んだParticle Gunによる2粒子発生ジェネレターを使い、磁場中での2粒子の運動をトラッキングしてみる – ここではPrimary GeneratorからGeometryに話題を少し寄り道をする
- 磁場の設定はジオメトリ記述で行えることを学ぶ

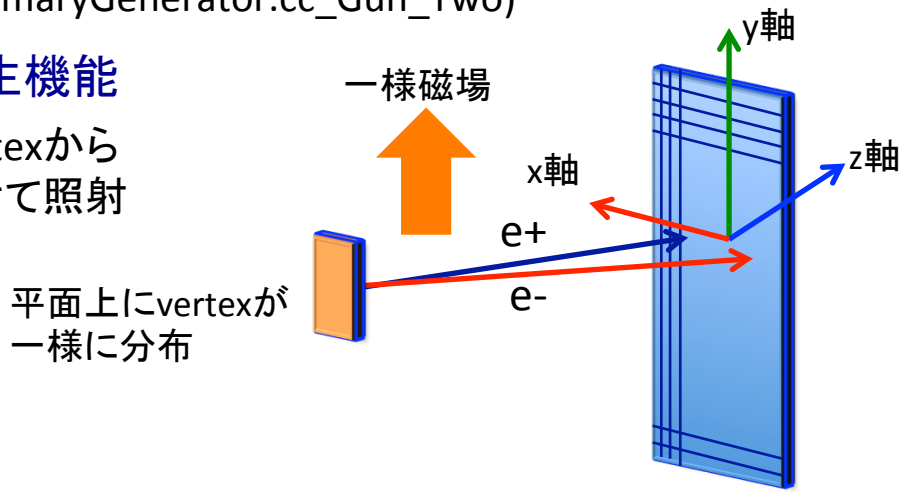
  ［注］この演習では、受講者はソースコードを書き、それを動かしてみる

■ プログラムの構成

- mainプログラム: P01_FirstStep_Visと同一
- ジオメトリ: P02_Geometryで使ったPixel検出器に磁場の設定を追加する
- PhysicsList : P01_FirstStep_Visと同一
- PrimaryGenerator: 前の演習と同じ (PrimaryGenerator.cc_Gun_Two)

■ 組み込まれているジオメトリと粒子発生機能

- 粒子発生: 平面に一様に分布するvertexから e+とe-の粒子をPixelに向けて照射



P03_PrimaryGenerator: MagField プログラムの作成

はじめに:

P03_PrimaryGenerator: MagFieldプログラムを作成するには:

1) プログラム作成作業はP03_PrimaryGenerator: Gun_Twoで使ったファイルをもとに行う
2) 本演習で変更が必要なファイルはGeometry.ccのみで、それ以外は全て流用

課題:1　P03_PrimaryGenerator: MagFieldの作業ファイルの用意

1) 現在のGeometry.ccファイルは前課題で使ったGun_Two用のもの: このファイルをエディトして"MagField"用のものを作成する
   [注] Geometry.hhは変更する必要はない
2) 念のため、オリジナルのP02_Geometry: Pixel_Oneの".cc"ファイルを現在のGeometry.ccに再度コピーしておく
   (Geometry.ccを変更した記憶がなければこのステップはスキップ可)

```
$ cd ~/Geant4Tutorial20171129/UserWorkDir/P03_PrimaryGenerator ← CBDir (Current
$ cd source/src                                                  Base Directory)
$ cp ../../../P02_Geometry/util/G4Codes/Geometry.cc_P02_Pixel_One Geometry.cc
overwrite Geometry.cc? (y/n [n]) y
```

P03_PrimaryGenerator: MagFieldの実装

課題：2　現在のGeometry.ccからの変更箇所を理解する

［注］ 以下のソースコードは変更が完了したものを示す

エディット手順は
次のスライド参照

```
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// Geometry.cc
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
#include "Geometry.hh"
#include "G4Box.hh"
#include "G4Tubs.hh"
#include "G4LogicalVolume.hh"
#include "G4PVPlacement.hh"
#include "G4PVReplica.hh"
#include "G4VPhysicalVolume.hh"
#include "G4ThreeVector.hh"
#include "G4RotationMatrix.hh"
#include "G4Transform3D.hh"
#include "G4NistManager.hh"
#include "G4VisAttributes.hh"
#include "G4SystemOfUnits.hh"
#include "G4GlobalMagFieldMessenger.hh"

//-----------------------------------------------------------------------------
  Geometry::Geometry() {}
//-----------------------------------------------------------------------------
```

追加

Geometry.ccファイルの
先頭部分

```
// Placement of the 'Pixcel Detector' to the world: Put the 'global envelop'
   G4double pos_X_LogV_PixEnvG = 0.0*cm;       // X-location LogV_PixEnvG
   G4double pos_Y_LogV_PixEnvG = 0.0*cm;       // Y-location LogV_PixEnvG
   G4double pos_Z_LogV_PixEnvG = 0.0*cm;       // Z-location LogV_PixEnvG
   G4ThreeVector threeVect_LogV_PixEnvG =
     G4ThreeVector(pos_X_LogV_PixEnvG, pos_Y_LogV_PixEnvG, pos_Z_LogV_PixEnvG);
   G4RotationMatrix rotMtrx_LogV_PixEnvG = G4RotationMatrix();
   G4Transform3D trans3D_LogV_PixEnvG = G4Transform3D(rotMtrx_LogV_PixEnvG, threeVect_LogV_PixEnvG);

   G4int copyNum_LogV_PixEnvG = 1000;              // Set ID number of LogV_PixEnvG
   new G4PVPlacement(trans3D_LogV_PixEnvG, "PhysVol_PixEnvG", logVol_PixEnvG, physVol_World,
                     false, copyNum_LogV_PixEnvG);

// Add uniform magnetic field to the world
   G4ThreeVector magFieldVector = G4ThreeVector(0.0, 1.0*tesla, 0.0);
   new G4GlobalMagFieldMessenger(magFieldVector);

// Return the physical world
   return physVol_World;
}
```

追加

Geometry.ccファイルの
最終部分

P03_PrimaryGenerator: MagFieldの実装 – つづき

課題：3　現在のGeometry.ccをエディットして磁場を実装する

.ccのあるdir

1）現在のGeometry.ccとMagField用（模範解答）の相違をcolordiffで端末に表示する

```
$ cd ~/Geant4Tutorial20171129/UserWorkDir/P03_PrirmaryGenerator/source/src
$ colordiff -y --width=200  Geometry.cc ../../util/G4Codes/Geometry.cc_MagField
```

Pixel_Oneファイル

```
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// Geometry.cc
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
#include "Geometry.hh"
#include "G4Box.hh"
#include "G4Tubs.hh"
#include "G4LogicalVolume.hh"
#include "G4PVPlacement.hh"
#include "G4PVReplica.hh"
#include "G4VPhysicalVolume.hh"
#include "G4ThreeVector.hh"
#include "G4RotationMatrix.hh"
#include "G4Transform3D.hh"
#include "G4NistManager.hh"
#include "G4VisAttributes.hh"
#include "G4SystemOfUnits.hh"


//-----------------------------------------------------------------------------
  Geometry::Geometry() {}
//-----------------------------------------------------------------------------
```

MagField追加ファイル

```
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// Geometry.cc
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
#include "Geometry.hh"
#include "G4Box.hh"
#include "G4Tubs.hh"
#include "G4LogicalVolume.hh"
#include "G4PVPlacement.hh"
#include "G4PVReplica.hh"
#include "G4VPhysicalVolume.hh"
#include "G4ThreeVector.hh"
#include "G4RotationMatrix.hh"
#include "G4Transform3D.hh"
#include "G4NistManager.hh"
#include "G4VisAttributes.hh"
#include "G4SystemOfUnits.hh"
#include "G4GlobalMagFieldMessenger.hh"

//-----------------------------------------------------------------------------
  Geometry::Geometry() {}
//-----------------------------------------------------------------------------
```

追加ライン印

途中部　表示省略

```
// Placement of the 'Pixel Detector' to the world: Put the 'global envelop'
   G4double pos_X_LogV_PixEnvG = 0.0*cm;       // X-location LogV_PixEnvG
   G4double pos_Y_LogV_PixEnvG = 0.0*cm;       // Y-location LogV_PixEnvG
   G4double pos_Z_LogV_PixEnvG = 0.0*cm;       // Z-location LogV_PixEnvG
   G4ThreeVector threeVect_LogV_PixEnvG =
     G4ThreeVector(pos_X_LogV_PixEnvG, pos_Y_LogV_PixEnvG, pos_Z_LogV_PixEnvG);
   G4RotationMatrix rotMtrx_LogV_PixEnvG = G4RotationMatrix();
   G4Transform3D trans3D_LogV_PixEnvG = G4Transform3D(rotMtrx_LogV_PixEnvG, threeVect_LogV_PixEn

   G4int copyNum_LogV_PixEnvG = 1000;              // Set ID number of LogV_PixEnvG
   new G4PVPlacement(trans3D_LogV_PixEnvG, "PhysVol_PixEnvG", logVol_PixEnvG, physVol_World,
                     false, copyNum_LogV_PixEnvG);




// Return the physical world
   return physVol_World;
}
```

追加ライン印

```
// Placement of the 'Pixel Detector' to the world: Put the 'global envelop'
   G4double pos_X_LogV_PixEnvG = 0.0*cm;       // X-location LogV_PixEnvG
   G4double pos_Y_LogV_PixEnvG = 0.0*cm;       // Y-location LogV_PixEnvG
   G4double pos_Z_LogV_PixEnvG = 0.0*cm;       // Z-location LogV_PixEnvG
   G4ThreeVector threeVect_LogV_PixEnvG =
     G4ThreeVector(pos_X_LogV_PixEnvG, pos_Y_LogV_PixEnvG, pos_Z_LogV_PixEnvG);
   G4RotationMatrix rotMtrx_LogV_PixEnvG = G4RotationMatrix();
   G4Transform3D trans3D_LogV_PixEnvG = G4Transform3D(rotMtrx_LogV_PixEnvG, threeVect_LogV_PixEn

   G4int copyNum_LogV_PixEnvG = 1000;              // Set ID number of LogV_PixEnvG
   new G4PVPlacement(trans3D_LogV_PixEnvG, "PhysVol_PixEnvG", logVol_PixEnvG, physVol_World,
                     false, copyNum_LogV_PixEnvG);

// Add uniform magnetic field to the world
   G4ThreeVector magFieldVector = G4ThreeVector(0.0, 1.0*tesla, 0.0);
   new G4GlobalMagFieldMessenger(magFieldVector);

// Return the physical world
   return physVol_World;
}
```

2）現在のGeometry.ccをエディターで開き、colordiffの結果を参照しながらMagFieldを実装する

```
$ pwd
$ gedit Geometry.cc&
```

Geometry.ccのあるdirであることを確認 – 異なれば移動

[注1]　変更をタイプするのが大変なら、以下のファイルをターミナルで表示して、変更箇所をコピペすること
　　$ less ../../util/G4Codes/Geometry.cc_MagField
[注2]　コピペがうまくできない場合、以下のコマンドを実行して模範解答を全コピーする
　　$ cp ../../util/G4Codes/Geometry.cc_MagField Geometry.cc

P03_PrimaryGenerator: MagFieldアプリケーションのビルド

課題：4　新たに作ったGeometry.ccを用いてアプリケーションをビルドする

1） 現在のベースデレクトリに移動

```
$ cd ~/Geant4Tutorial20171129/UserWorkDir/P03_PrimaryGenerator
```

← CBDir (Current Base Directory)

2) buildデレクトリでビルドを実行　[注1]

```
$ \rm -r bin
$ cd  build
$ \rm -r *
$ cmake ../source
$ make
$ make install
$ cd ..
```

← 前のプログラムをビルドした時に作らたbinは消去（省略可）

← 前のプログラムをビルドした時に作られたbuildに移動

← 前のビルドで作られているファイル類を全て消去

cmakeの慣用句

[注1]
buildのステップでerrorが出て、どうしても
アプリケーションが作れない場合、CBdirのもとで
以下のスクリプトを実行すれば、模範解答の
PrimaryGenerator:GPSをもとにbuildを
自動完了することができる

./util/Help/Build_P03_MagField.sh

Application_Main (MagField)の実行

課題：5　Application_Main (MagField)を実行する

1) プログラム実行を行う作業デレクトリに移動

```
$ pwd
……/P03_PrimaryGenerator
$ cd TestBench
$ ../bin/Application_Main
```

現デレクトリがCBDirであることを確認：
P03_PrimaryGeneratorでなければ、そこへ移動

作業デレクトリに移動

TestBenchデレクトリでApplication_Main実行

2) 端末ウインドに以下のメッセージが出力され、続いてQtウインドが開く

```
**************************************************************
 Geant4 version Name: geant4-10-03-patch-02    (20-October-2017)
                      Copyright : Geant4 Collaboration
                      Reference : NIM A 506 (2003), 250-303
                            WWW : http://cern.ch/geant4
**************************************************************

<<< Geant4 Physics List simulation engine: FTFP_BERT 2.0
.....
### Adding tracking cuts for neutron  TimeCut(ns)= 10000  KinEnergyCut(MeV)= 0
Visualization Manager instantiating with verbosity "warnings (3)"...
Visualization Manager initialising...
Registering graphics systems...
.....
```

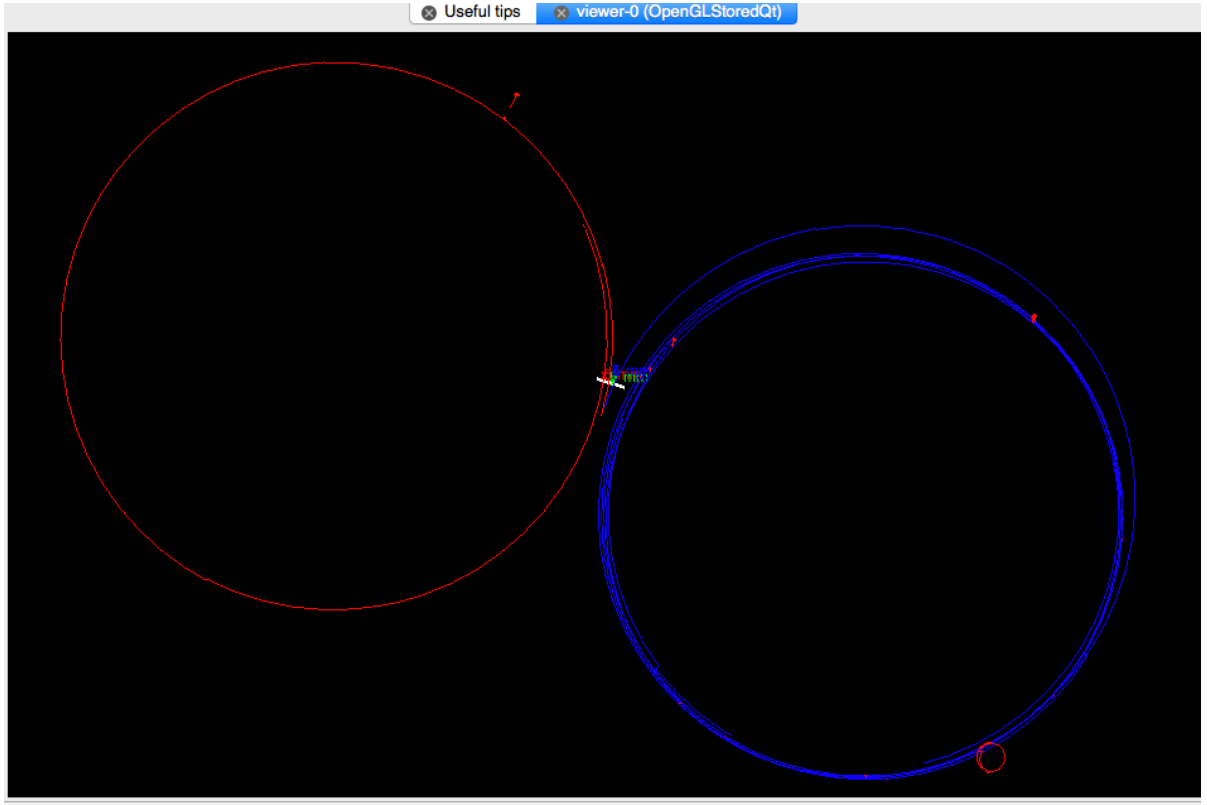
Qtウインドでアプリの動作を確認

課題:6　UI コマンドで事象発生

1) 以下のUIコマンドを先ず入力

```
/run/beamOn 1
```

右図はその出力
（マウスで視点を調整してある）



2) マウスで視点を変えてみる

3) 照射回数を自由に変更してみる

4) アプリ終了

```
exit
```

付録

P03_PrimaryGenerator: MyGen_One

ユーザが独自の初期粒子定義をしたい場合

P03_PrimaryGenerator: MyGen_Oneプログラムの概要

■ 演習プログラムの目的

- 今までの演習では初期粒子発生にG4ParticleGun及びG4GeneralParticleSourceというGeant4が標準で提供するツールを使った。ここでは、ユーザが独自のPrimary Generatorを作成する方法を学ぶ
- まずは単純に粒子を一つだけ発生させるgeneratorを作り、作成の基礎を学ぶ

［注］この演習では、用意されているプログラムを編集することなく、そのまま使う

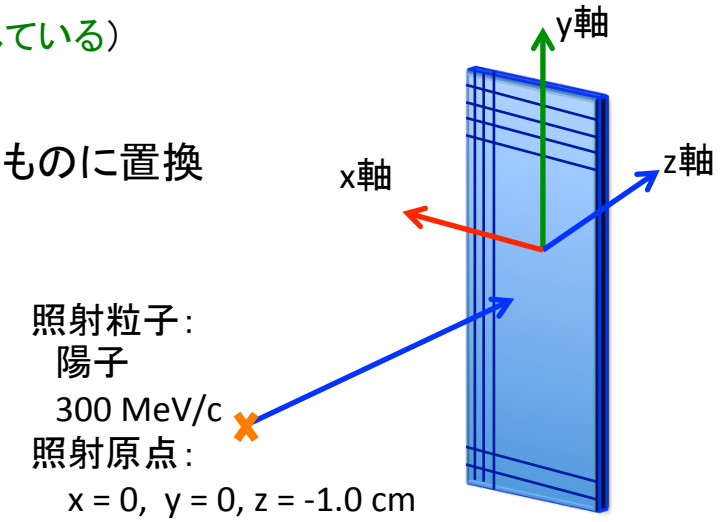
■ プログラムの構成

- mainプログラム： P01_FirstStep_Visと同一
- ジオメトリ： P02_Geometryで使ったPixel検出器
  （直前の課題で追加した磁場が含まれている）
- PhysicsList : P01_FirstStep_Visと同一
- PrimaryGenerator: 前演習のコードを提供されているものに置換

[注] このプログラム構成は前演習の
「P03_PrimaryGenerator: MagField」を終了した
状態と同一であることを仮定している

■ 組み込まれているジオメトリと粒子発生機能

- 粒子発生： 点源からの発生



P03_PrimaryGenerator: MyGen_Oneのビルド

課題：1　P03_PrimaryGenerator: MyGen_Oneをビルドして実行ファイルを作成する

1）ソースファイルPrimaryGenerator.cc_MyGen_OneをPrimaryGenerator.ccにコピー

```
$ cd ~/Geant4Tutorial20171129/UserWorkDir/P03_PrimaryGenerator
$ cd source/src
$ cp ../../util/G4Codes/PrimaryGenerator.cc_MyGen_One PrimaryGenerator.cc
overwrite PrimaryGenerator.cc? (y/n [n]) y
```

CBDir (Current Base Directory)

2）ヘッダファイルPrimaryGenerator.hh_MyGem_OneをPrimaryGenerator.hhにコピー

```
$ cd ../include
$ cp ../../util/G4Codes/PrimaryGenerator.hh_MyGen_One PrimaryGenerator.hh
overwrite PrimaryGenerator.hh? (y/n [n]) y
```

3）buildデレクトリでビルドを実行　[注1]

```
$ cd ../..
$ \rm -r bin
$ cd  build
$ \rm -r *
$ cmake ../source
$ make
$ make install
```

CBDir (Current Base Directory)へ戻る

前のプログラムをビルドした時に作られたbuildに移動

前のビルドで作られているファイル類を全て消去

前のプログラムをビルドした時に作らたbinは消去（省略可）

cmakeの慣用句

[注1]
buildを失敗したら、CBdirのもとで以下のスクリプトを実行すればbuildは自動完了
./util/Help/Build_P03_MyGen_One.sh

Application_Main (MyGen_One)の実行

課題：2　Application_Main (MyGen_One)を実行する

1) プログラム実行を行う作業デレクトリに移動

```
$ pwd                                                    ← 現デレクトリがCBDirであることを確認：
……/P03_PrimaryGenerator                                   P03_PrimaryGeneratorでなければ、そこへ移動
$ cd TestBench                                           ← 作業デレクトリに移動
$ cp ../util/Macros_PrimaryGen/primaryGeneratorSetup.mac_MyGen_One \
     primaryGeneratorSetup.mac                           ← MyGen用のprimaryGeneratorSetup.macをコピー
overwrite primaryGeneratorSetup.mac? (y/n [n]) y
$ ../bin/Application_Main                                ← TestBenchデレクトリでApplication_Main実行
```

2) 端末ウインドに以下のメッセージが出力され、続いてQtウインドが開く

```
*************************************************************
 Geant4 version Name: geant4-10-03-patch-03    (20-October-2017)
                      Copyright : Geant4 Collaboration
                      Reference : NIM A 506 (2003), 250-303
                            WWW : http://cern.ch/geant4
*************************************************************

<<< Geant4 Physics List simulation engine: FTFP_BERT 2.0
.....
### Adding tracking cuts for neutron  TimeCut(ns)= 10000  KinEnergyCut(MeV)= 0
Visualization Manager instantiating with verbosity "warnings (3)"...
Visualization Manager initialising...
Registering graphics systems...
.....
```

Qtウインドでアプリの動作を確認

課題:3　UI コマンドで事象発生

1) 以下のUIコマンドを先ず入力

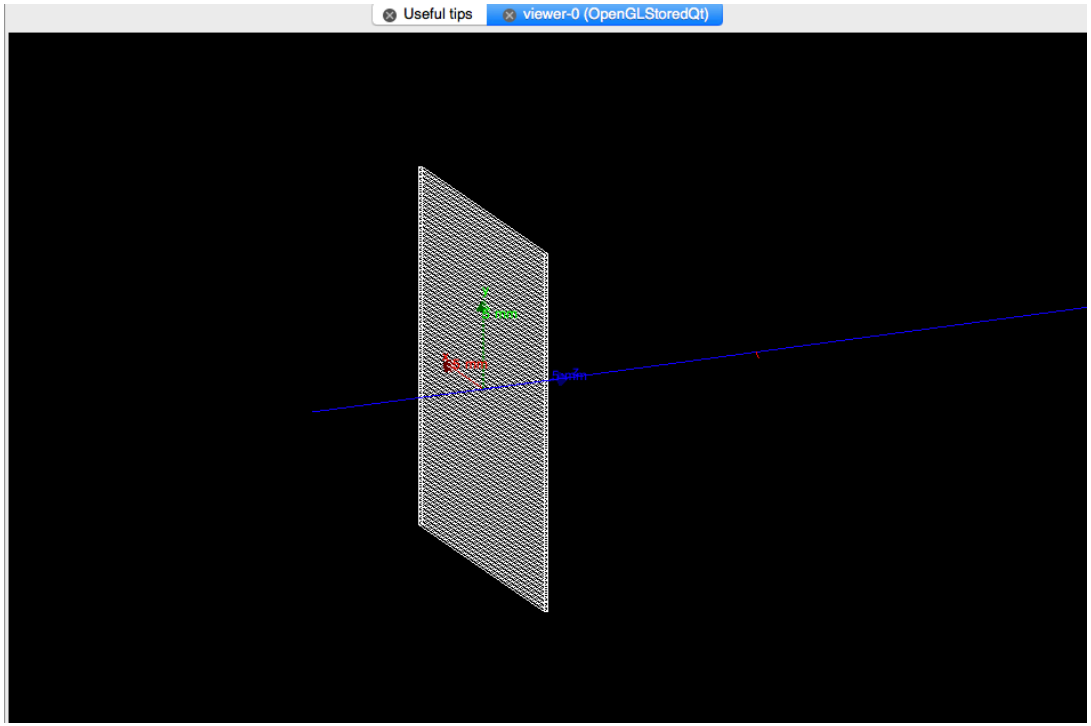
```
/run/beamOn 1
```

右図はその出力
（マウスで視点を調整してある）

2) 照射回数を自由に変更してみる

3) アプリ終了

```
exit
```



（Geometry: MagFieldの磁場あり）

P03_PrimaryGenerator: MyGen_OneのPrmimaryGenerator.cc/.hhファイルの内容

課題:5　ヘッダー及び実装ファイルを理解する

```
$ less ../source/include/PrimaryGenerator.hh
$ less ../source/src/PrimaryGenerator.cc
```

ヘッダーファイル

```
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// PrimaryGenerator.hh
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
#ifndef PrimaryGenerator_h
#define PrimaryGenerator_h 1

#include "G4VUserPrimaryGeneratorAction.hh"
#include "G4ThreeVector.hh"
class G4Event;
class G4ParticleDefinition;

//---------------------------------------------------------------------------
  class PrimaryGenerator : public G4VUserPrimaryGeneratorAction
//---------------------------------------------------------------------------
{
  public:
    PrimaryGenerator();
   ~PrimaryGenerator();

  public:
    void GeneratePrimaries(G4Event*);
};
#endif
```

この関数を実装

実装ファイル

```
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// PrimaryGenerator.cc
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
#include "PrimaryGenerator.hh"
#include "G4Event.hh"
#include "G4PrimaryVertex.hh"
#include "G4PrimaryParticle.hh"
#include "G4ParticleTable.hh"
#include "G4SystemOfUnits.hh"

//---------------------------------------------------------------------------
  PrimaryGenerator::PrimaryGenerator()
//---------------------------------------------------------------------------
{}

//---------------------------------------------------------------------------
  PrimaryGenerator::~PrimaryGenerator()
//---------------------------------------------------------------------------
{}

//---------------------------------------------------------------------------
  void PrimaryGenerator::GeneratePrimaries(G4Event* anEvent)
//---------------------------------------------------------------------------
{
// Particle table
   G4ParticleTable* particleTable = G4ParticleTable::GetParticleTable();

// Setup for primary particle
   G4String particleName = "proton";
   G4double momentum = 300.0*MeV;
   G4ThreeVector momentumVector = G4ThreeVector(0.0, 0.0, 1.0)*momentum;

// Create a primary particle - need to create for every event
   G4PrimaryParticle* primaryParticle =
     new G4PrimaryParticle( particleTable->FindParticle(particleName),
                            momentumVector.x(), momentumVector.y(), momentumVector.z() );

// Create a primary vertex - need to create for every event
   G4ThreeVector vertex = G4ThreeVector( 0.0*cm, 0.0*cm, -1.0*cm);
   G4PrimaryVertex* primaryVertex = new G4PrimaryVertex(vertex, 0.0*second);

// Add the primary particles to the primary vertex
   primaryVertex->SetPrimary( primaryParticle );

// Add the vertex to the event
   anEvent->AddPrimaryVertex( primaryVertex );
}
```

primary particleをnewで生成
（事象発生ごとに毎回new）

primary vertexをnewで生成
（事象発生ごとに毎回new）

primary particleをvertexに配置

primary vertexをeventに付加

P03_PrimaryGenerator: MyGen_Two

P03_PrimaryGenerator: Gun_Twoと同じ機能の実装

P03_PrimaryGenerator: MyGen_Twoプログラムの概要

■ 演習プログラムの目的

- G4ParticleGunを使った2粒子発生と同じ機能をユーザ独自のPrimary Generatorで作成する方法を学ぶ

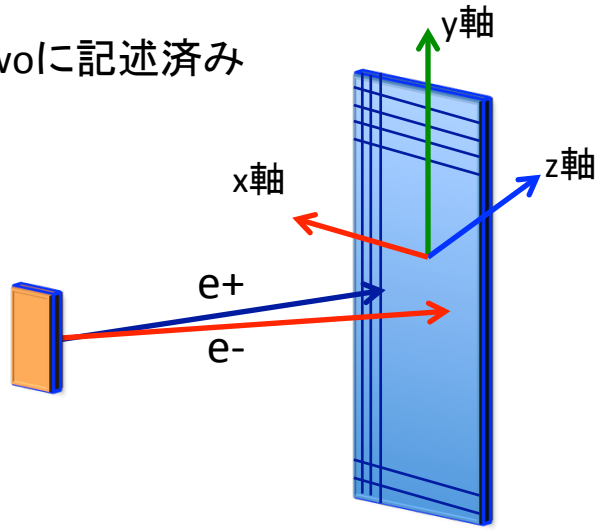
［注］この演習では、用意されているプログラムを編集することなく、そのまま使う

■ プログラムの構成

- mainプログラム： P01_FirstStep_Visと同一
- ジオメトリ： P02_Geometryで使ったPixel検出器
  （MagFieldの磁場あり）
- PhysicsList： P01_FirstStep_Visと同一
- PrimaryGenerator: PrimaryGenerator.cc_MyGen_Twoに記述済み

■ 組み込まれているジオメトリと粒子発生機能

- 平面PrimaryGenerator.cc_Gun_Twoと全く同一の構成が実装されている



平面上にvertexが一様に分布

P03_PrimaryGenerator: MyGen_Twoのビルド

課題：1　P03_PrimaryGenerator: MyGen_Twoをビルドして実行ファイルを作成する

1）ソースファイルPrimaryGenerator.cc_MyGen_TwoをPrimaryGenerator.ccにコピー

```
$ cd ~/Geant4Tutorial20171129/UserWorkDir/P03_PrimaryGenerator
$ cd source/src
$ cp ../../util/G4Codes/PrimaryGenerator.cc_MyGen_Two PrimaryGenerator.cc
overwrite PrimaryGenerator.cc? (y/n [n]) y
```

CBDir (Current Base Directory)

2）ヘッダファイルPrimaryGenerator.hh_MyGem_TwoをPrimaryGenerator.hhにコピー

```
$ cd ../include
$ cp ../../util/G4Codes/PrimaryGenerator.hh_MyGen_Two PrimaryGenerator.hh
overwrite PrimaryGenerator.hh? (y/n [n]) y
```

3）buildデレクトリでビルドを実行　[注1]

```
$ cd ../..
$ \rm -r bin
$ cd  build
$ \rm -r *
$ cmake ../source
$ make
$ make install
```

CBDir (Current Base Directory)へ戻る
前のプログラムをビルドした時に作らたbinは消去（省略可）
前のプログラムをビルドした時に作られたbuildに移動
前のビルドで作られているファイル類を全て消去

cmakeの慣用句

[注1]
buildを失敗したら、CBdirのもとで以下のスクリプトを実行すればbuildは自動完了
./util/Help/Build_P03_MyGen_Two.sh

Application_Main (MyGen_Two)の実行

課題：2　Application_Main (MyGen_Two)を実行する

1) プログラム実行を行う作業デレクトリに移動

```
$ pwd                                                    ← 現デレクトリがCBDirであることを確認：
……/P03_PrimaryGenerator                                   P03_PrimaryGeneratorでなければ、そこへ移動
$ cd TestBench                                           ←
$ cp ../util/Macros_PrimaryGen/primaryGeneratorSetup.mac_MyGen_Two \
     primaryGeneratorSetup.mac                           ← MyGen用のprimaryGeneratorSetup.macをコピー
overwrite primaryGeneratorSetup.mac? (y/n [n]) y
$ ../bin/Application_Main                                ← TestBenchデレクトリでApplication_Main実行
```

2) 端末ウインドに以下のメッセージが出力され、続いてQtウインドが開く

```
**************************************************************
 Geant4 version Name: geant4-10-03-patch-03    (20-October-2017)
                      Copyright : Geant4 Collaboration
                      Reference : NIM A 506 (2003), 250-303
                            WWW : http://cern.ch/geant4
**************************************************************

<<< Geant4 Physics List simulation engine: FTFP_BERT 2.0
.....
### Adding tracking cuts for neutron  TimeCut(ns)= 10000  KinEnergyCut(MeV)= 0
Visualization Manager instantiating with verbosity "warnings (3)"...
Visualization Manager initialising...
Registering graphics systems...
.....
```

Qtウインドでアプリの動作を確認

課題:3　UI コマンドで事象発生

1) 以下のUIコマンドを先ず入力

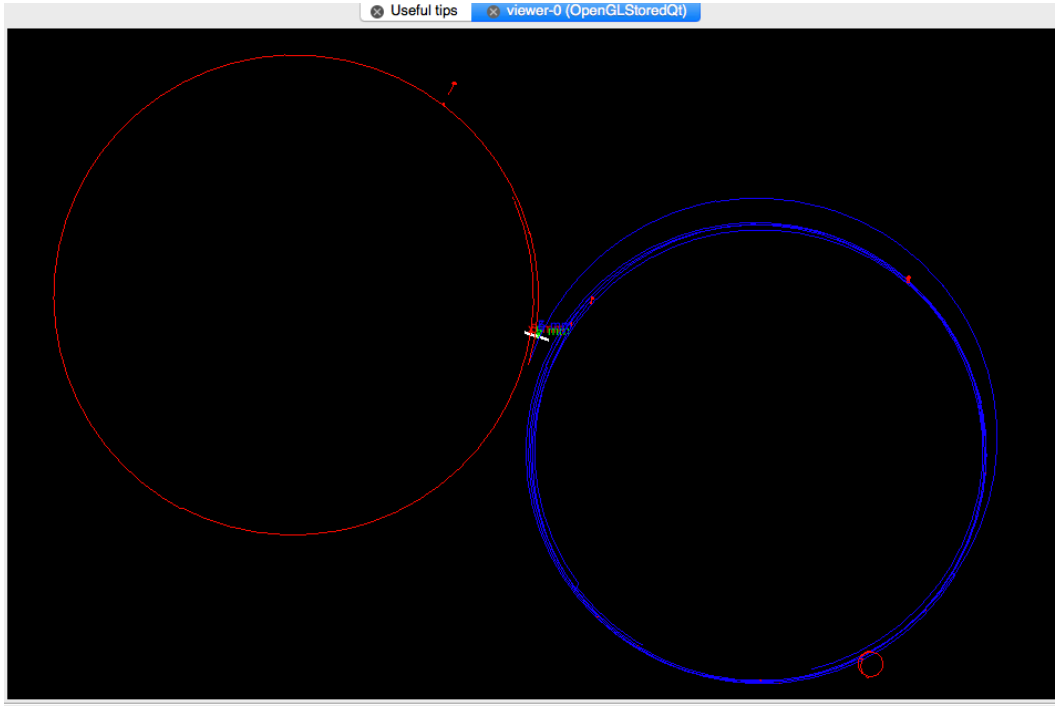
```
/run/beamOn 1
```

右図はその出力
（マウスで視点を調整してある）

2) 照射回数を自由に変更してみる

3) アプリ終了

```
exit
```



（Geometry: MagFieldの磁場あり）

P03_PrimaryGenerator: MyGen_TwoのPrmimaryGenerator.hhファイルの内容

課題:5　初期粒子ヘッダー・ファイルを理解する

```
$ less ../source/include/PrimaryGenerator.hh
```

```
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// PrimaryGenerator.hh
//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
#ifndef PrimaryGenerator_h
#define PrimaryGenerator_h 1

#include "G4VUserPrimaryGeneratorAction.hh"
#include "G4ThreeVector.hh"
class G4Event;
class G4ParticleDefinition;

//---------------------------------------------------------------------------
  class PrimaryGenerator : public G4VUserPrimaryGeneratorAction
//---------------------------------------------------------------------------
{
  public:
    PrimaryGenerator();
   ~PrimaryGenerator();

  public:
    void GeneratePrimaries(G4Event*);

  private:
    G4ParticleDefinition* fParDef_1;        // Partilce definition: particle #1
    G4ThreeVector fMomVect_1;               // Momentum vector: particle #1
    G4ParticleDefinition* fParDef_2;        // Partilce definition: particle #2
    G4ThreeVector fMomVect_2;               // Momentum vector: particle #2
};
#endif
```

この関数を実装

粒子発射条件の保存

P03_PrimaryGenerator: MyGen_TwoのPrmimaryGenerator.ccファイルの内容

課題:6　初期粒子実装ファイルを理解する

```
$ less ../source/src/PrimaryGenerator.cc
```

```
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
// PrimaryGenerator.cc
//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
#include "PrimaryGenerator.hh"
#include "G4Event.hh"
#include "G4PrimaryVertex.hh"
#include "G4PrimaryParticle.hh"
#include "G4ParticleTable.hh"
#include "G4SystemOfUnits.hh"
#include "Randomize.hh"

//---------------------------------------------------------------------------
  PrimaryGenerator::PrimaryGenerator()
//---------------------------------------------------------------------------
{
// Particle table
   G4ParticleTable* particleTable = G4ParticleTable::GetParticleTable();

// 1st primary particle - setup for fixed parameters
   {
     G4String particleName = "e-";
     G4double momentum = 50.0*MeV;
     G4double angle = 1.0*deg;
     G4ThreeVector momentumDirection = G4ThreeVector(0.0, 0.0, 1.0).rotateY(angle);
     fMomVect_1 = momentumDirection*momentum;
     fParDef_1  = particleTable->FindParticle(particleName);
   }

// 2ne primary particle - setup for fixed parameters
   {
     G4String particleName = "e+";
     G4double momentum = 50.0*MeV;
     G4double angle = -1.0*deg;
     G4ThreeVector momentumDirection = G4ThreeVector(0.0, 0.0, 1.0).rotateY(angle);
     fMomVect_2 = momentumDirection*momentum;
     fParDef_2  = particleTable->FindParticle(particleName);
   }
}

//---------------------------------------------------------------------------
  PrimaryGenerator::~PrimaryGenerator()
//---------------------------------------------------------------------------
{}
```

必要なヘッダー・ファイルのinclude

一番目の粒子の発射条件（固定値）

一番目の粒子発射条件の設定

二番目の粒子の発射条件（固定値）

二番目の粒子発射条件の設定

次のスライドにコードはつづく

P03_PrimaryGenerator: MyGen_TwoのPrmimaryGenerator.ccファイルの内容

課題:6　初期粒子実装ファイルを理解する（つづき）

```
//....oooOO0OOooo.....oooOO0OOooo.....oooOO0OOooo.....oooOO0OOooo.....oooOO0OOooo....
  void PrimaryGenerator::GeneratePrimaries(G4Event* anEvent)
//....oooOO0OOooo.....oooOO0OOooo.....oooOO0OOooo.....oooOO0OOooo.....oooOO0OOooo....
{

// Create primary particles - need to create for every event
   G4PrimaryParticle* primaryParticle_1 =
     new G4PrimaryParticle( fParDef_1, fMomVect_1.x(), fMomVect_1.y(), fMomVect_1.z() );
   G4PrimaryParticle* primaryParticle_2 =
     new G4PrimaryParticle( fParDef_2, fMomVect_2.x(), fMomVect_2.y(), fMomVect_2.z() );

// Primary vertex  position - randomization
   G4double pos_X = 2.0*mm*(G4UniformRand()-0.5);
   G4double pos_Y = 2.0*mm*(G4UniformRand()-0.5);
   G4double pos_Z = -2.0*cm;
   G4ThreeVector vertex = G4ThreeVector(pos_X, pos_Y, pos_Z);
   G4double time_Zero = 0.0*ns;

// Create a primary vertex - need to create for every event
   G4PrimaryVertex* primaryVertex = new G4PrimaryVertex(vertex, time_Zero);

// Add the primary particles to the primary vertex
   primaryVertex->SetPrimary( primaryParticle_1 );
   primaryVertex->SetPrimary( primaryParticle_2 );

// Add primary vertext to the event
   anEvent->AddPrimaryVertex( primaryVertex );
}
```

primary particleを newで生成 （事象発生ごとに毎回new）

vertex位置を乱数で決定

primary vertexを newで生成 （事象発生ごとに毎回new）

二つのprimary particleを vertexに配置

primary vertexを eventに付加

終了