Hokkaido 2023
Geant4 training school in medicine

Primary Generator Action

Geant4 Online documentation

David Bolst - dbolst@uow.edu.au
University of Wollongong

Primary Generator Action

- The **PrimaryGeneratorAction** is one of the three required classes for a Geant4 simulation
- Derive your concrete class from **G4VUserPrimaryGeneratorAction** abstract base class
  - Geant4 provides several generators in addition to the G4VPrimaryParticlegenerator base class
    - **G4ParticleGun**
    - **G4GeneralParticleSource**
    - **G4HEPEvtInterface, G4HepMCInterface**
- Pass a G4Event object to one or more primary generator concrete class objects which generate primary vertices/primary particles (parentID =0),

Typical User Primary Generator Action

//Contents of PrimaryGeneratorAction.cc

```
#include "PrimaryGeneratorAction.hh"
#include "G4Event.hh"
#include "G4ParticleGun.hh"
#include "G4GeneralParticleSource.hh"

PrimaryGeneratorAction::PrimaryGeneratorAction()
{
//For using the Particle Gun
gun  = new G4ParticleGun();


//Or For general particle source (GPS)
GPSgun  = new G4GeneralParticleSource();
}

PrimaryGeneratorAction::~PrimaryGeneratorAction()
{
delete gun;
//Or For general particle source (GPS)
delete GPSgun;
}
void PrimaryGeneratorAction::GeneratePrimaries(G4Event*
anEvent)
{...}
```

//Contents of PrimaryGeneratorAction.hh

```
#ifndef PrimaryGeneratorAction_hh
#define PrimaryGeneratorAction_hh 1

#include "G4VUserPrimaryGeneratorAction.hh"
#include "G4Event.hh"
#include "globals.hh"

class G4GeneralParticleSource;
class G4ParticleGun;
class G4Event;

class PrimaryGeneratorAction : public
G4VUserPrimaryGeneratorAction {

public:
PrimaryGeneratorAction();
~PrimaryGeneratorAction();
void GeneratePrimaries(G4Event*);

private:
G4GeneralParticleSource* GPSgun;
//Or
G4ParticleGun* gun;
};

#endif
```

Particle Gun and General Particle Source (GPS)

- The two main methods to generate primary particles in Geant4 are
  - The “General Particle Source” (GPS)
  - The “Particle Gun”

GPS	Particle Gun
-GPS is declared in the user’s Primary Generator Action class but all options of the source is set with UI commands (usually in macro files) -GPS offers ability to create many different distributions using built-in commands	-Particle Gun is declared in the user’s action class with the source’s characteristics being set within the class itself -More flexible and powerful, but can be much more time consuming than doing equivalent action with the GPS

Particle gun or GPS?

- For new users, the GPS is a good starting point
  - GPS usually offers faster and simpler ways of implementing sources
  - Lots of different options, though care should be taken when generating sources with GPS as they have many commands whose default can behave unexpectedly (example later)
- Certain tasks, such as using a phase space file (PSF-more on this later), require the use of a G4 Particle Gun
- Particle gun can be very built to fit a specific application, for instance arguments taken in the main() may be used to set source location, shape, energy, type more seamlessly than possible with the GPS and UI commands

Setting particle attributes with GPS

- The General Particle Source (GPS) provides commands which can be used in macros
  - No compiling of program is required for changing source setting within macro
- Provides ability to generate complex particle distributions with simple commands

GPS Online Documentation

Lots of GPS examples!

Selection of *some* GPS commands

Table 2 G4ParticleGun equivalent commands.¶

Command	Arguments	Description and restrictions
/gps/List		List available incident **particles**
/gps/particle	name	Defines the **particle** type [default *geantino*], using GEANT4 naming convention.
/gps/direction	Px Py Pz	Set the momentum direction [default (1,0,0)] of generated **particles** using (1)
/gps/energy	E unit	Sets the energy [default 1 MeV] for mono-energetic **sources**. The units can be eV, keV, MeV, GeV, TeV or PeV. (NB: it is recommended to use /gps/ene/mono instead.)
/gps/position	X Y Z unit	Sets the centre co-ordinates (X,Y,Z) of the **source** [default (0,0,0) cm]. The units can be micron, mm, cm, m or km. (NB: it is recommended to use /gps/pos/centre instead.)
/gps/ion	Z A Q E	After /gps/**particle** ion, sets the properties (atomic number Z, atomic mass A, ionic charge Q, excitation energy E in keV) of the ion.
/gps/ionLvl	Z A Q lvl	After /gps/**particle** ion, sets the properties (atomic number Z, atomic mass A, ionic charge Q, Number of metastable state excitation level (0-9) of the ion.
/gps/time	t0 unit	Sets the primary **particle** (event) time [default 0 ns]. The units can be ps, ns, us, ms, or s.
/gps/polarization	Px Py Pz	Sets the polarization vector of the **source**, which does not need to be a unit vector.
/gps/number	N	Sets the number of **particles** [default 1] to simulate on each event.
/gps/verbose	level	Control the amount of information printed out by the GPS code. Larger values produce more detailed output.

Setting particle attributes with Particle Gun

- The user must implement their own methods to generate source distributions (angular, energy, field size)
- Some attributes of ParticleGun can be controlled via UI commands such as:
  - /gun/List
  - /gun/particle particlename
  - /gun/energy ## MeV
  - /gun/direction # #

```
//G4ParticleGun Class Reference (excluding Get
functions)
G4ParticleGun()
G4ParticleGun(G4int numberofparticles)
G4ParticleGun(G4ParticleDefinition *particleDef, G4int
numberofparticles = 1)
virtual ~G4ParticleGun()
virtual void GeneratePrimaryVertex(G4Event *evt)

void SetParticleDefinition(G4ParticleDefinition
*aParticleDefinition)
void SetParticleEnergy(G4double aKineticEnergy)
void SetParticleMomentum(G4double aMomentum)
void SetParticleMomentum(G4ParticleMomentum aMomentum)
void SetParticleMomentumDirection(G4ParticleMomentum
aMomentumDirection)
void SetParticleCharge(G4double aCharge)
void SetParticlePolarization(G4ThreeVector aVal)
void SetNumberOfParticles(G4int i)
```

Some examples for creating sources with GPS and the particle gun

- Sources are generated in a 1x1x1 m^3 cube filled with "G4_Galactic" (vacuum)
- Executing program and macro:

```
./programName macroName.mac
```

GPS

```
//In the PrimaryGeneratorAction.cc
void PrimaryGeneratorAction::GeneratePrimaries(G4Event*
anEvent)
{GPSgun->GeneratePrimaryVertex(anEvent);}
```

```
#Contents of macro file
/physics/addPhysics QGSP_BIC_EMY #add some physics
/run/initialize
/control/verbose 0
#GPS commands here
/tracking/verbose 1 #track info outputted for
these examples
#Shoot X number of particles
/run/beamOn 10
```

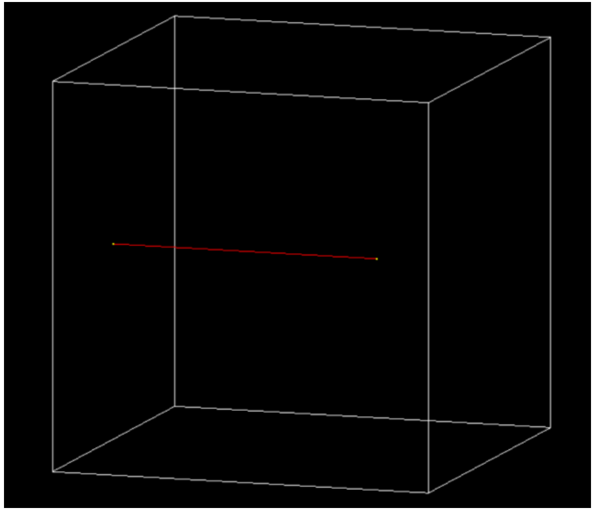
Particle Gun

```
//In the PrimaryGeneratorAction.cc
void PrimaryGeneratorAction::GeneratePrimaries(G4Event*
anEvent)
{//Particle gun commands here
gun->GeneratePrimaryVertex(anEvent);}
```

```
#Contents of macro file
/physics/addPhysics QGSP_BIC_EMY #add some physics
/run/initialize
/control/verbose 0
/tracking/verbose 1 #track info outputted for these
examples
#Shoot X number of particles
/run/beamOn 10
```

Producing an electron pencil beam

- Electron beam is generated at Z = 200 mm with a mono-energetic energy of 100 keV and travels in the –Z direction
  - For visualising, by default negative charged particles are red, positive are blue and neutron are green, steps (interactions boundaries) are yellow



```
*******************************************************************************************
* G4Track Information:   Particle = e-,   Track ID = 1,   Parent ID = 0
*******************************************************************************************
Step#    X(mm)    Y(mm)    Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
    0        0        0      200       0.1        0        0         0 physicalWorld initStep
    1        0        0     -500       0.1  4.7e-23      700       700  OutOfWorld Transportation
*******************************************************************************************
* G4Track Information:   Particle = e-,   Track ID = 1,   Parent ID = 0
*******************************************************************************************
Step#    X(mm)    Y(mm)    Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
    0        0        0      200       0.1        0        0         0 physicalWorld initStep
    1        0        0     -500       0.1  4.7e-23      700       700  OutOfWorld Transportation
*******************************************************************************************
* G4Track Information:   Particle = e-,   Track ID = 1,   Parent ID = 0
*******************************************************************************************
Step#    X(mm)    Y(mm)    Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
    0        0        0      200       0.1        0        0         0 physicalWorld initStep
    1        0        0     -500       0.1  4.7e-23      700       700  OutOfWorld Transportation
```

Producing electron pencil beam with GPS

- Note: When using the GPS for your simulation, you still must generate the vertex within your code (**PrimaryGeneratorAction** for instance)

```
//In the PrimaryGeneratorAction.cc
void PrimaryGeneratorAction::GeneratePrimaries(G4Event*
anEvent)
{GPSgun->GeneratePrimaryVertex(anEvent);}
```

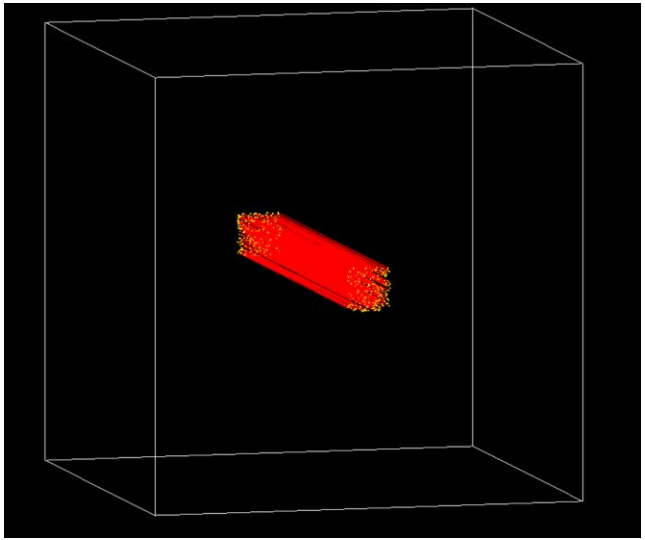
```
#Commands in macro
#Select particle type
/gps/particle e-

#Set beam's energy
/gps/ene/mono 100 keV

#Defining the Source Shape
/gps/pos/type Point

#Set beam's momentum
/gps/direction 0 0 -1

#Set beam's starting position
/gps/pos/centre 0. 0. 20. cm
```

Producing electron pencil beam with Particle Gun

```
//In the PrimaryGeneratorAction.cc
void PrimaryGeneratorAction::GeneratePrimaries(G4Event* anEvent)
{
//Set the particle type
G4ParticleDefinition* particle=0;
particle = G4ParticleTable::GetParticleTable()->FindParticle("e-");
gun->SetParticleDefinition(particle);

//Set the beam energy
G4double energy = 100.*keV;
gun->SetParticleEnergy(energy);

//Set the beam direction
G4double xMomentum = 0.;
G4double yMomentum = 0.;
G4double zMomentum = -1.; //Going in negative Z-direction
gun->SetParticleMomentumDirection(G4ThreeVector(xMomentum,yMomentum,zMomentum));

//Set the starting beam position
G4double startX = 0.;
G4double startY = 0.;
G4double startZ = 20.*cm;
gun->SetParticlePosition(G4ThreeVector(startX, startY,  startZ));

gun->GeneratePrimaryVertex(anEvent);
}
```

Creating a 10x10 cm^2 beam

- Similar as before, e- beam is generated at Z = 500 mm, travelling in the negative Z-direction with 100 keV energy but the field size is 100 x 100 mm^2



```
**************************************************************************************************
* G4Track Information:   Particle = e-,   Track ID = 1,   Parent ID = 0
**************************************************************************************************
Step#    X(mm)    Y(mm)    Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
   0     -37     11.8      500       0.1       0       0        0 physicalWorld initStep
   1     -37     11.8     -500       0.1 6.71e-23    1e+03    1e+03  OutOfWorld Transportation
**************************************************************************************************
* G4Track Information:   Particle = e-,   Track ID = 1,   Parent ID = 0
**************************************************************************************************
Step#    X(mm)    Y(mm)    Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
   0   -38.1     15.6      500       0.1       0       0        0 physicalWorld initStep
   1   -38.1     15.6     -500       0.1 6.71e-23    1e+03    1e+03  OutOfWorld Transportation
**************************************************************************************************
* G4Track Information:   Particle = e-,   Track ID = 1,   Parent ID = 0
**************************************************************************************************
Step#    X(mm)    Y(mm)    Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
   0    3.45       41      500       0.1       0       0        0 physicalWorld initStep
   1    3.45       41     -500       0.1 6.71e-23    1e+03    1e+03  OutOfWorld Transportation
```

Creating a 10x10 cm^2 beam

GPS

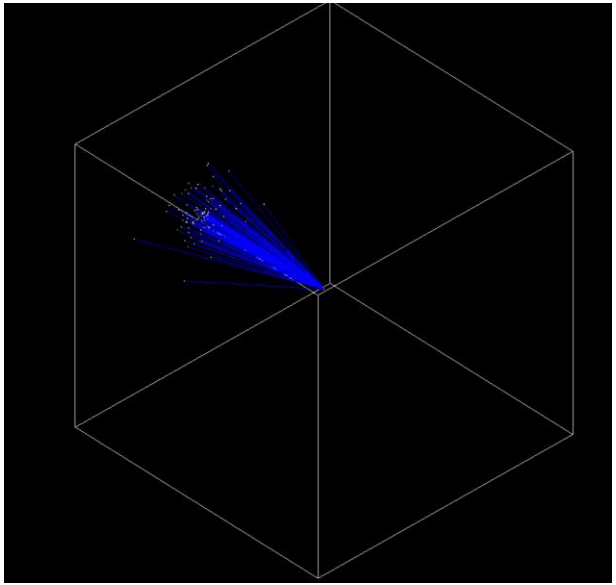
```
#Defining the Source Shape
/gps/pos/shape Square
/gps/pos/type Beam
/gps/pos/halfx 5 cm
/gps/pos/halfy 5 cm

#Set beam's momentum
/gps/direction 0 0 -1

#Set beam's starting position
/gps/pos/centre 0. 0. 50. cm
```

Particle Gun

```
//Set the starting beam position
G4double startX = 0.;
G4double startY = 0.;
G4double startZ = 50.*cm;

G4double sizeX = 100.*mm;
G4double sizeY = 100.*mm;

//Generate distribution from -0.5 and 0.5 and scale by size
of beam
startX = (sizeX * (G4UniformRand() - 0.5));
startY = (sizeY * (G4UniformRand() - 0.5));

gun->SetParticlePosition(G4ThreeVector(startX, startY,
startZ));
gun->SetParticleMomentumDirection(G4ThreeVector(0, 0, -1));
```

Creating a “diverging” beam

- When firing a proton or heavy ion beam (or any accelerator based beam) the beam will have some divergence and shape
  - For this example we’ll shoot a 290 MeV/u carbon beam
  - Beam will diverge by 10 degrees (large amount of divergence for a for a real hadron therapy beam)
  - Will have a Gaussian energy distribution of 1 sigma (bit large for carbon, typically ~0.002)



```
*********************************************************************************************
* G4Track Information:   Particle = C12,   Track ID = 1,   Parent ID = 0
*********************************************************************************************
Step#    X(mm)    Y(mm)    Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
   0       0       0       0  3.45e+03       0       0       0 physicalWorld initStep
   1   -29.6    25.9    -500  3.45e+03 9.36e-22     502     502  OutOfWorld Transportation
*********************************************************************************************
* G4Track Information:   Particle = C12,   Track ID = 1,   Parent ID = 0
*********************************************************************************************
Step#    X(mm)    Y(mm)    Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
   0       0       0       0  3.53e+03       0       0       0 physicalWorld initStep
   1   -31.8   -0.427    -500  3.53e+03 9.23e-22     501     501  OutOfWorld Transportation

*********************************************************************************************
* G4Track Information:   Particle = C12,   Track ID = 1,   Parent ID = 0
*********************************************************************************************
Step#    X(mm)    Y(mm)    Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
   0       0       0       0  3.48e+03       0       0       0 physicalWorld initStep
   1   -65.3   -40.4    -500  3.48e+03  9.4e-22     506     506  OutOfWorld Transportation
```

Creating a “diverging” beam

GPS

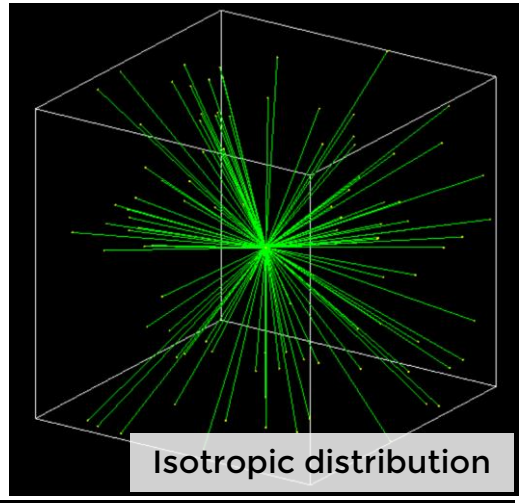
```
/gps/particle ion
/gps/ion 6 12 6

/gps/ene/type Gauss
#Use total energy of ion 290 MeV/u * 12 u
/gps/ene/mono 3480. MeV
#Applying a 1% sigma -> totalEnergy * 0.01
/gps/ene/sigma 34.8 MeV

/gps/pos/type Beam
/gps/ang/type beam1d
/gps/ang/sigma_r 10. deg

/gps/pos/centre 0. 0. 0. mm

        #To have Gaussian size
        profile 5 mm sigma
        /gps/pos/shape Circle
        /gps/pos/radius 0. mm
        /gps/pos/sigma_r 5 mm
```

Particle Gun

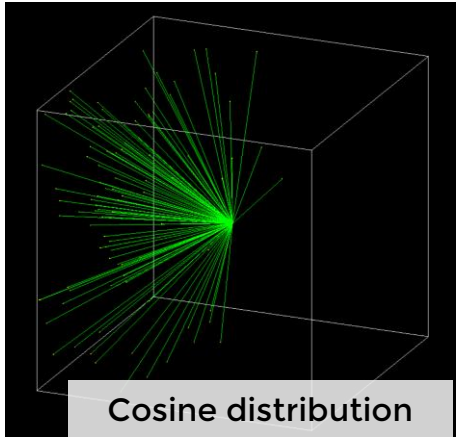
```
G4ParticleDefinition* particle = 0;
particle = G4IonTable::GetIonTable()->GetIon(6, 12, 0);
gun->SetParticleDefinition(particle);
//Angle dist
G4double theta, phi;
G4double px, py, pz;
G4double sigmaAngle = 10.;//degrees
theta = G4RandGauss::shoot(0.0, (sigmaAngle*pi) / 180.);
phi = twopi * G4UniformRand();
px = -std::sin(theta) * std::cos(phi);
py = -std::sin(theta) * std::sin(phi);
pz = -std::cos(theta);
gun->SetParticleMomentumDirection(G4ThreeVector(px, py, pz));
gun->SetParticlePosition(G4ThreeVector(0, 0, 0));
//Energy
G4double beamEnergy = 12.*290.*MeV;
G4double energySigma = 0.01;
G4double finalEnergy = G4RandGauss::shoot(beamEnergy,
beamEnergy*energySigma);
gun->SetParticleEnergy(finalEnergy);
gun->GeneratePrimaryVertex(anEvent);
```

Creating an isotropic point source

- Geantinos shown being generated at the centre of the world (0,0,0) and is directed isotropically
  - A “geantino” is a pseudo-particle which has no physical interactions and is commonly used for testing geometry and beam distributions
- Note: when creating “isotropic” distributions for space applications, a cosine angular distribution should be adopted. See: G Santin 2007 for details



Isotropic distribution



Cosine distribution

```
**********************************************************************************************************
* G4Track Information:   Particle = geantino,   Track ID = 1,   Parent ID = 0
**********************************************************************************************************
Step#    X(mm)    Y(mm)    Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
   0       0        0        0    0.001       0       0        0 physicalWorld initStep
   1     324     -500     -436    0.001       0     739      739  OutOfWorld Transportation
**********************************************************************************************************
* G4Track Information:   Particle = geantino,   Track ID = 1,   Parent ID = 0
**********************************************************************************************************
Step#    X(mm)    Y(mm)    Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
   0       0        0        0    0.001       0       0        0 physicalWorld initStep
   1     500     -109     -418    0.001       0     660      660  OutOfWorld Transportation
**********************************************************************************************************
* G4Track Information:   Particle = geantino,   Track ID = 1,   Parent ID = 0
**********************************************************************************************************
Step#    X(mm)    Y(mm)    Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
   0       0        0        0    0.001       0       0        0 physicalWorld initStep
   1     434      297     -500    0.001       0     726      726  OutOfWorld Transportation
```

Creating an isotropic point source

GPS

```
/gps/particle geantino
/gps/pos/type Point
/gps/ang/type iso
/gps/pos/centre 0 0 0 m
```

Particle Gun

```
particle = G4ParticleTable::GetParticleTable()-
>FindParticle("geantino");
G4double twopi = 6.28318530718;
G4double cosTheta = -1.0 + 2.0*G4UniformRand();
G4double phi = twopi * G4UniformRand();
G4double sinTheta = sqrt(1 - cosTheta * cosTheta);

// these are the cosines for an isotropic direction
gun->SetParticleMomentumDirection(G4ThreeVector(sinTheta*cos(phi),
sinTheta*sin(phi), cosTheta));
gun->SetParticlePosition(G4ThreeVector(startX, startY, startZ));
```

Creating a cosine point source

GPS

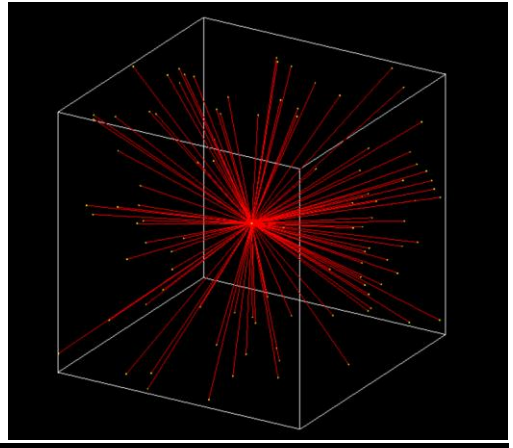
```
/gps/pos/type Point
/gps/ang/type cos
/gps/pos/centre 0 0 0 m
```

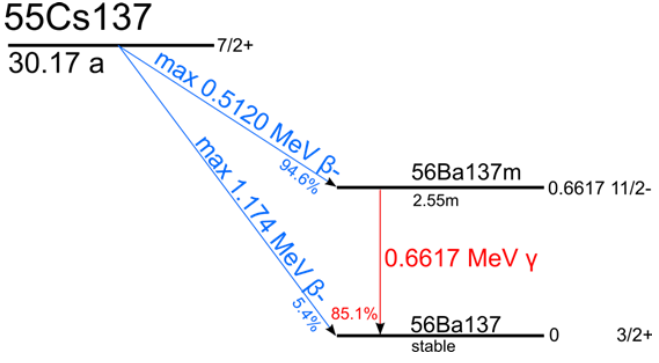
Particle Gun

```
//From G4SPSAngDistribution.cc
G4double MaxTheta = pi / 2.;G4double MaxPhi = 2.*pi;
G4double MinPhi = 0.; G4double MinTheta = 0.;
// Method to generate flux distributed with a cosine law
G4double px, py, pz;
G4double rndm, rndm2;
G4double sintheta, sinphi, costheta, cosphi;
rndm = G4UniformRand();
sintheta = std::sqrt(rndm * (std::sin(MaxTheta)*std::sin(MaxTheta) -
std::sin(MinTheta)*std::sin(MinTheta)) + std::sin(MinTheta)*std::sin(MinTheta));
costheta = std::sqrt(1. - sintheta * sintheta);

rndm2 = G4UniformRand();
G4double Phi = MinPhi + (MaxPhi - MinPhi) * rndm2;
sinphi = std::sin(Phi);
cosphi = std::cos(Phi);
px = -sintheta * cosphi;
py = -sintheta * sinphi;
pz = -costheta;
gun->SetParticleMomentumDirection(G4ThreeVector(px, py, pz));
gun->SetParticlePosition(G4ThreeVector(0, 0, 0));
gun->SetParticleEnergy(1.*keV);
gun->GeneratePrimaryVertex(anEvent);
```

Generating discrete energy distribution

- May wish to generate discrete energies such as gammas/x-rays released during a certain decay/de-excitation
- For this example using maximum energy of electrons emitted during the decay of Cs-137 and their corresponding weights





Cs137 decay

```
*********************************************************************************************
* G4Track Information:   Particle = e-,   Track ID = 1,   Parent ID = 0
*********************************************************************************************
Step#   X(mm)   Y(mm)   Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
   0      0       0      0      0.51       0        0        0 physicalWorld initStep
   1   -256     500     109     0.51 1.73e-23     572      572  OutOfWorld Transportation
*********************************************************************************************
* G4Track Information:   Particle = e-,   Track ID = 1,   Parent ID = 0
*********************************************************************************************
Step#   X(mm)   Y(mm)   Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
   0      0       0      0      0.51       0        0        0 physicalWorld initStep
   1    236     407    -500     0.51 2.08e-23     687      687  OutOfWorld Transportation
*********************************************************************************************
* G4Track Information:   Particle = geantino,   Track ID = 1,   Parent ID = 0
*********************************************************************************************
Step#   X(mm)   Y(mm)   Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
   0      0       0      0      1.17       0        0        0 physicalWorld initStep
   1      0       0   -500      1.17       0      500      500  OutOfWorld Transportation
```

Generating discrete energy distribution

GPS

```
#...
/gps/particle e-
/gps/ang/type iso
#Setting weight of first energy
/gps/source/intensity 0.946
/gps/ene/mono 0.51 MeV
#Setting weight of second energy
/gps/source/add 0.054
/gps/ene/mono 1.17 MeV
#...
```

Particle Gun

```
//...
if (G4UniformRand() < 0.946)
electronEnergy = 0.51*MeV
else
electronEnergy = 1.17*MeV
//...
```

Generating energy distribution

- Often you will have a energy spectrum such as photons of a LINAC or on-board imager which you may wish to perform some investigation with
- Shown here is a 6 MV LINAC X-ray

```
**************************************************************************************************
* G4Track Information:  Particle = gamma,  Track ID = 1,  Parent ID = 0
**************************************************************************************************
Step#   X(mm)   Y(mm)   Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
   0     -37   11.8     400    0.382       0       0        0 physicalWorld initStep
   1     -37   11.8    -500    0.382       0     900      900  OutOfWorld Transportation
**************************************************************************************************
* G4Track Information:  Particle = gamma,  Track ID = 1,  Parent ID = 0
**************************************************************************************************
Step#   X(mm)   Y(mm)   Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
   0    4.8    -36.3    400    0.343       0       0        0 physicalWorld initStep
   1    4.8    -36.3   -500    0.343       0     900      900  OutOfWorld Transportation
**************************************************************************************************
* G4Track Information:  Particle = gamma,  Track ID = 1,  Parent ID = 0
**************************************************************************************************
Step#   X(mm)   Y(mm)   Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
   0    7.28    12.4    400    1.44        0       0        0 physicalWorld initStep
   1    7.28    12.4   -500    1.44        0     900      900  OutOfWorld Transportation
```

6 MV LINAC energy distribution

```
/gps/hist/type arb
/gps/hist/point 0    0
/gps/hist/point 0.046439628 24.08256881
/gps/hist/point 0.13003096  165.1376147
/gps/hist/point 0.185758514 280.9633028
/gps/hist/point 0.232198142 353.2110092
/gps/hist/point 0.297213622 411.6972477
/gps/hist/point 0.343653251 444.9541284
/gps/hist/point 0.473684211 457.5688073
/gps/hist/point 0.659442724 434.6330275
/gps/hist/point 0.761609907 412.8440367
/gps/hist/point 0.835913313 392.2018349
/gps/hist/point 0.956656347 370.412844
/gps/hist/point 1.049535604 346.3302752
/gps/hist/point 1.160990712 324.5412844
/gps/hist/point 1.263157895 299.3119266
/gps/hist/point 1.458204334 266.0550459
/gps/hist/point 1.681114551 232.7981651
/gps/hist/point 1.885448916 207.5688073
/gps/hist/point 2.191950464 174.3119266
/gps/hist/point 2.470588235 145.6422018
/gps/hist/point 2.767801858 126.146789
/gps/hist/point 3.074303406 105.5045872
/gps/hist/point 3.371517028 87.1559633
/gps/hist/point 3.650154799 75.68807339
/gps/hist/point 4.114551084 57.33944954
/gps/hist/point 4.59752322  41.28440367
/gps/hist/point 5.00619195  27.52293578
/gps/hist/point 5.470588235 16.05504587
/gps/hist/point 5.990712074 2.293577982
```

Generating energy distribution

GPS

```
/gps/particle gamma
/gps/pos/shape Square
/gps/pos/type Beam
/gps/pos/halfx 5 cm
/gps/pos/halfy 5 cm

#Implementing the gamma Energy Spectrum
#GPS will normalise provided weights
/gps/ene/type Arb
/gps/hist/type arb
/gps/hist/point 0 0
/gps/hist/point 0.0464396282 4.08256881
/gps/hist/point 0.1300309616 5.1376147
#..
/gps/hist/point 5.990712074 2.293577982
/gps/hist/inter Lin
```

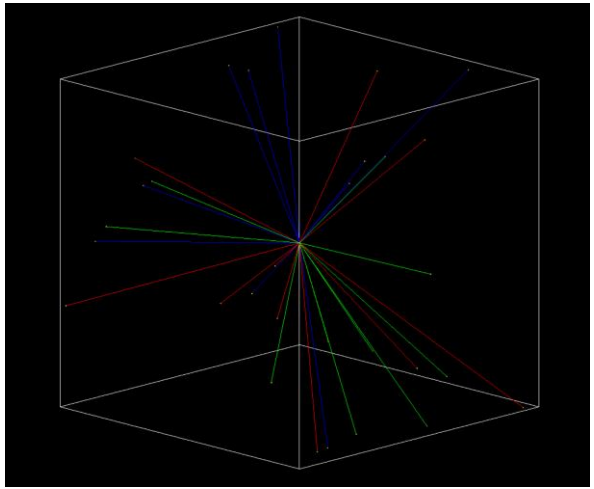
Energy Weight

Particle Gun

```
//Cumulative distribution in an array
G4double cumEnergyDist[distSize];
G4double Energy[distSize];
//..
G4double randCum = G4UniformRand();

for (int dd = 0; dd < distSize; dd++)
{
      if (randCum > cumEnergyDist[dd])
      {
      gun->SetParticleEnergy(Energy[dd]);
      break;
      }
}
//..
```

Generating point source isotopes for decay

- May wish to generate radioactive isotopes such as for brachytherapy or radiation shielding of sources
- Generating an isotope directly within the simulation will allow for an accurate decay chain as opposed to generating only certain energies in the decay



```
*********************************************************************************************************
* G4Track Information:   Particle = Cs137,   Track ID = 1,   Parent ID = 0
*********************************************************************************************************
Step#    X(mm)    Y(mm)    Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
   0       0       0       0       0       0       0       0 physicalWorld initStep
   1       0       0       0       0       0       0       0 physicalWorld RadioactiveDecay
*********************************************************************************************************
* G4Track Information:   Particle = e-,   Track ID = 4,   Parent ID = 1
*********************************************************************************************************
Step#    X(mm)    Y(mm)    Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
   0       0       0       0   0.463       0       0       0 physicalWorld initStep
   1    -101     396    -500   0.463 2.01e-23     646     646  OutOfWorld Transportation
*********************************************************************************************************
* G4Track Information:   Particle = anti_nu_e,   Track ID = 3,   Parent ID = 1
*********************************************************************************************************
Step#    X(mm)    Y(mm)    Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
   0       0       0       0  0.0513       0       0       0 physicalWorld initStep
   1    -254     500     455  0.0513       0     722     722  OutOfWorld Transportation
*********************************************************************************************************
* G4Track Information:   Particle = Ba137[661.659],   Track ID = 2,   Parent ID = 1
*********************************************************************************************************
Step#    X(mm)    Y(mm)    Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
   0       0       0       0  2.7e-06       0       0       0 physicalWorld initStep
   1     122    -446     500       0  2.7e-06     681     681 physicalWorld Transportation
```

Generating isotopes for decay

GPS

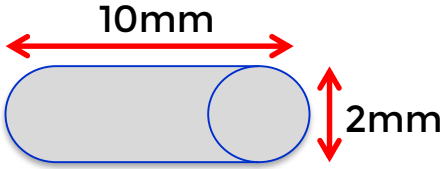
```
//...
/gps/particle ion
/gps/ion 55 137 55
/gps/energy 0 MeV
//...
```

Particle Gun

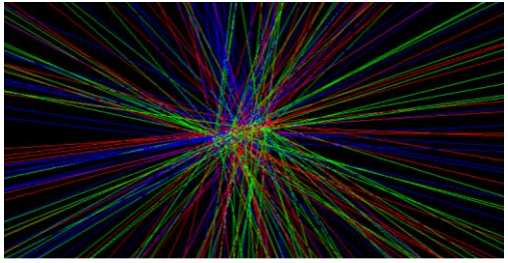
```
//...
#include "G4ParticleDefinition.hh"
#include "G4ParticleTable.hh"
#include "G4IonTable.hh"
//...
particle = G4IonTable::GetIonTable()-
>GetIon(27, 60, 0);
gun->SetParticleDefinition(particle);
gun->SetParticleEnergy(0);
```

Generating a 3D source

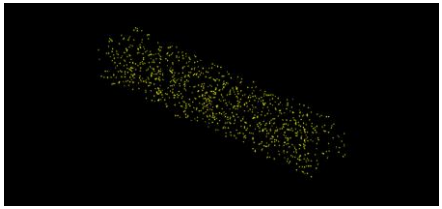
- May wish to simulate a bulk radioactive material such as a brachytherapy seed
- How to model a 10x2 mm cylinder Cs137 seed



BXTA



Generating 100x Cs137



Generating 1000x primaries and killing secondaries to check shape

```
**************************************************************************************************
* G4Track Information:   Particle = Cs137,   Track ID = 1,   Parent ID = 0
**************************************************************************************************
Step#    X(mm)    Y(mm)    Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
   0       0       0       0       0       0       0       0 physicalWorld initStep
   1       0       0       0       0       0       0       0 physicalWorld RadioactiveDecay
**************************************************************************************************
* G4Track Information:   Particle = e-,   Track ID = 4,   Parent ID = 1
**************************************************************************************************
Step#    X(mm)    Y(mm)    Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
   0       0       0       0   0.463       0       0       0 physicalWorld initStep
   1    -101     396    -500   0.463 2.01e-23     646     646  OutOfWorld Transportation
**************************************************************************************************
* G4Track Information:   Particle = anti_nu_e,   Track ID = 3,   Parent ID = 1
**************************************************************************************************
Step#    X(mm)    Y(mm)    Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
   0       0       0       0  0.0513       0       0       0 physicalWorld initStep
   1    -254     500     455  0.0513       0     722     722  OutOfWorld Transportation
**************************************************************************************************
* G4Track Information:   Particle = Ba137[661.659],   Track ID = 2,   Parent ID = 1
**************************************************************************************************
Step#    X(mm)    Y(mm)    Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
   0       0       0       0  2.7e-06       0       0       0 physicalWorld initStep
   1     122    -446     500       0  2.7e-06     681     681 physicalWorld Transportation
```

Generating source within a volume

GPS

```
/gps/particle ion
/gps/ion 55 137 55 0.
/gps/energy 0. keV
/gps/pos/type Volume
/gps/pos/shape Cylinder
/gps/pos/radius 1. mm
/gps/pos/halfz 5 mm
/gps/pos/centre 0. 0. 0. mm
```

Particle Gun

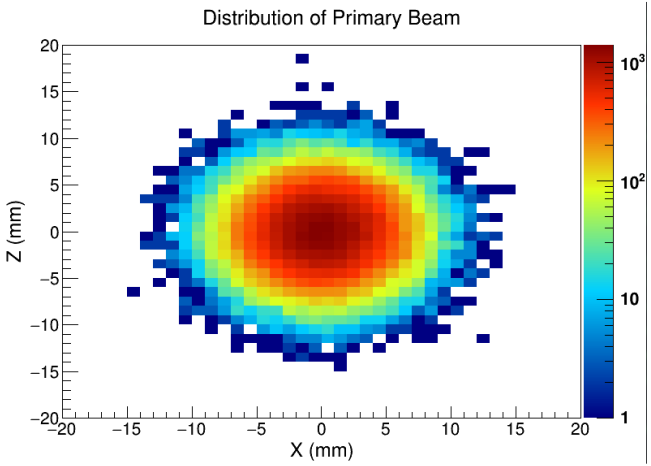
```
//Seed dimensions
G4double seedRadius = 1.*mm;
G4double seedLength = 10.*mm;
//Seed positions
G4double seedPosX = 0;
G4double seedPosY = 0;
G4double seedPosZ = 0;

//Generate random point on the cross-section of the rod
G4double PI = 3.14159265359;
G4double theta = 2.*PI*G4UniformRand();
G4double randomR = sqrt(seedRadius * seedRadius *
G4UniformRand());
startX = seedPosX + randomR * cos(theta);
startY = seedPosY + randomR * sin(theta);

//Generate random point along the length of seed
G4double randomSeedLengthPos =
(seedLength)*G4UniformRand() - seedLength/2.;
startZ = seedPosZ + randomSeedLengthPos;
//...
```

Careful: Checking your distributions (1)

- Check that the modelled source is what you expect it to be,
- by checking with /tracking/verbose 1 (first check)
- storing particle information in a histogram or ntuple and analyse the results

- Example, when shooting beams using the GPS a Z-direction beam is typically assumed

```
/gps/pos/shape Circle
/gps/pos/type Beam
/gps/pos/radius 0. mm
/gps/pos/sigma_r 3.4 mm
/gps/pos/centre 0. 0 0. cm
/gps/direction 0 -1 0
```

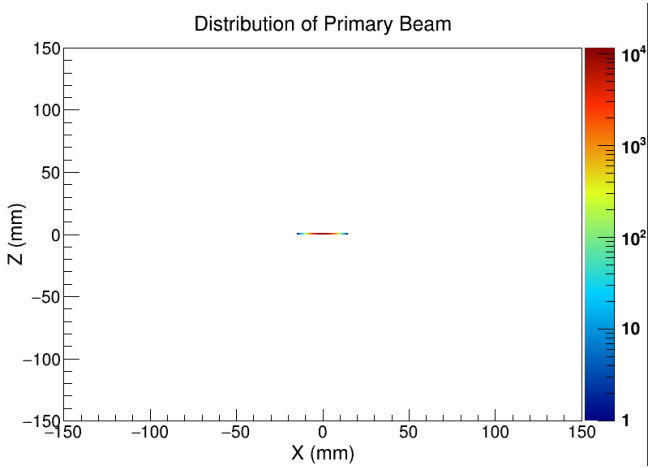


Expected?

Careful: Checking your distributions (2)

- Though running the previous commands would result in something like shown on the right
- To get the “expected” distribution you would need to rotate the beam into the xz plane using the commands in red below

```
/gps/pos/shape Circle
/gps/pos/type Beam
/gps/pos/radius 0. mm
/gps/pos/sigma_r 3.4 mm
/gps/pos/centre 0. 0 0. cm
/gps/direction 0 -1 0
```



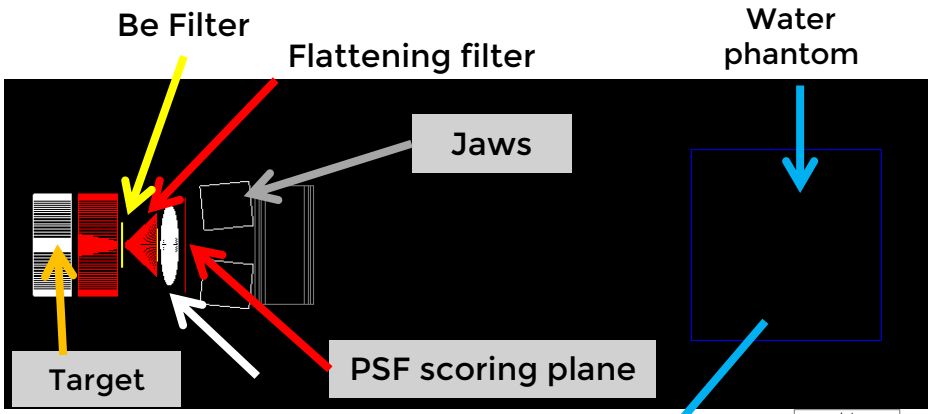
Actual

```
#Missing link

#Rotate beam to be shot onto the xz plane
/gps/pos/rot1 1 0 0
/gps/pos/rot2 0 0 1
```

Using phase space files

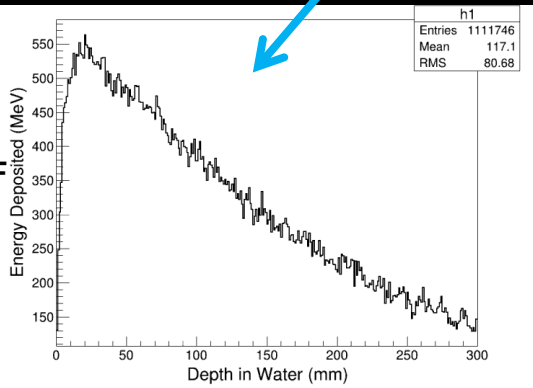
- Phase space files (PSFs) are a collection of particle:
  - Type (e.g. photon, electron)
  - Position
  - Momentum
  - Energy
- PSFs can result in significant speedup
- PSFs are common in medical physics, especially for LINAC
- Loading in your own phase space file

Simplified LINAC geometry



~15 hours **without** PSF

<70 second **with** PSF



IAEA phase space files

https://www-nds.iaea.org/phsp/phsp.htmlx

International Atomic Energy Agency
Nuclear Data Services
قسم البيانات النووية مقدمة من

International Advisory Committee (IAC)
R. Jeraj
I. Kawrakow
C.-M. Ma
D.W.O. Rogers
F. Sanchez-Doblado
J. Sempau
J. Seuntjens
J.V. Siebers
P. Andreo

Mailing Lists
Send mail to all members of the IAC
Register to the IAEA PHSP mailing list

Medical Portal
Atomic and nuclear data for medical applications

IAEA NAPC/NDS
Nuclear Data Section

IAEA NAHU/DMRP

Dosimetry and Medical Radiation Physics Section

Phase-space database for external beam radiotherapy

IAEA NAPC Nuclear Data Section
IAEA NAHU Dosimetry and Medical Radiation Physics Section

Project Officer: Roberto Capote

Objective: To build a database and disseminate representative phase-space data of accelerators and Co-60 units used in medical radiotherapy by compiling existing data that have been properly validated.

NEWS

How to produce and submit phase-space data: The IAEA phsp format was designed to cover both phase-space files and event generators (see phsp contents). We have implemented the IAEA phsp format in a set of read/write routines (Updated: September 2013, see readme file). Native IAEA phsp format is available in EGSnrc and PENELOPE Monte Carlo codes. Geant4 interface to use the native IAEA phsp format is also available. Unfortunately, MCNP does not have a native implementation of the IAEA format. A generic converter from IAEA to ASCII is also available as iaea2ascii.zip. Once the validated phsp data is produced and documentation is published, you may submit your phsp for review using the upload link here.

How to download phase-space data: You have to select a phsp data type among Co-60 source, linac electron or linac photon phsps. For photon and electron PHSPs you may download the header first to decide which data you want to retrieve. Once decided you should download the PHSP data from the corresponding sub-directory. Please note that the first time access to the selected subdirectory could be slow.

Both the PHSP data and header should be present for the PHSP data to be accesible !

In exceptional cases, you may request a DVD copy of the desired phsp.

Tech. Report
IAEA-NDS-0484

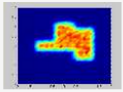
PHSP format
List of PHSP variables

PHSP Header
How to fill header ...

PHSP upload
Upload files

PHSP to review
Files to review

PHSP database



1. Co-60 phsps
2. Photon phsps
3. Electron phsps

https://www-nds.iaea.org/phsp/photon1/

Phase-space database for external beam radiotherapy

Project Officer: Roberto Capote

List of photon PHSP data for linear accelerators

Name	Last modified	Size	Description
Parent Directory		-	
CyberKnife_IRIS/	2011-10-06 14:07	-	
ELEKTA_Precise_10MV/	2011-10-06 14:09	-	
ELEKTA_Precise_25MV/	2011-10-06 14:09	-	
ELEKTA_Precise_6MV/	2011-10-06 14:10	-	
SIEMENS_Primus_6MV/	2011-10-06 14:10	-	
Varian_Clinac_600C_6MV/	2011-10-06 14:10	-	
Varian_Clinac_iX_6MV/	2011-10-06 14:11	-	
Varian_TrueBeam_6MV/	2011-10-06 14:11	-	
CyberKnife_IRIS_10mm.IAEAheader	2011-07-27 08:46	2.0K	
CyberKnife_IRIS_15mm.IAEAheader	2011-07-27 08:46	2.0K	
CyberKnife_IRIS_5mm.IAEAheader	2011-07-27 08:46	2.0K	
CyberKnife_IRIS_60mm_part1.IAEAheader	2011-07-27 08:46	2.0K	
CyberKnife_IRIS_60mm_part2.IAEAheader	2011-07-27 08:46	2.0K	
CyberKnife_IRIS_60mm_part3.IAEAheader	2011-07-27 08:46	2.0K	

Summary of steps generating and reading PSF

- Generating the PSF involves:
  - Set your desired beam configuration
  - Creating a “dummy volume” in the detector construction positioned in a “strategic location”
  - If a particle enters this volume, then the particles details will be stored in a file

- Reading in PSF involves:
  - Reading in PSF using the Primary Generator class

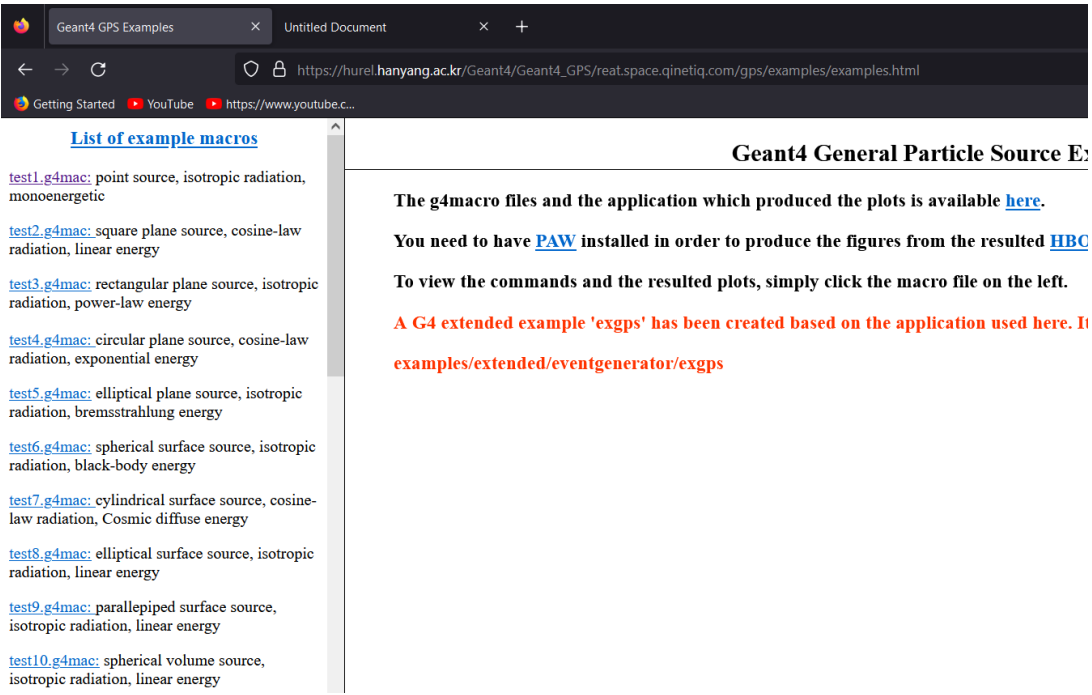
- See an example in the “Extra slides” to how you may generate and read your own PSF as well as more info on using the IAEA PSF format

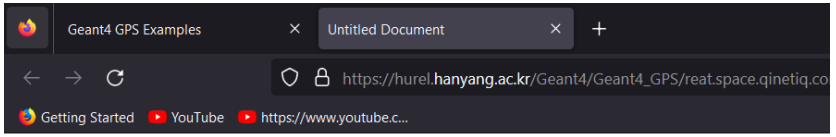
Questions?

Extra slides

GPS Examples

- https://hurel.hanyang.ac.kr/Geant4/Geant4_GPS/reat.space.qinetiq.com/gps/examples/examples.html
  - Very useful!

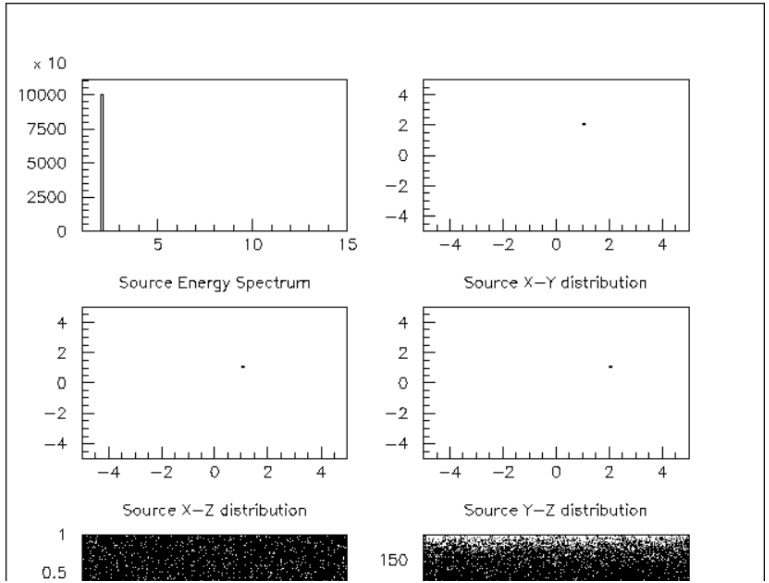




GPS Command Example 1:

```
/gps/particle proton
/gps/pos/type Point
/gps/pos/centre 1. 2. 1. cm
/gps/ang/type iso
/gps/energy 2. MeV
```

Resulted Distribution Plots:



View GPS source code

- https://ecce-eic.github.io/doxygen/d7/dbc/G4SPSAngDistribution_8cc_source.html
- https://ecce-eic.github.io/doxygen/d2/d12/G4SPSPosDistribution_8cc_source.html
- https://ecce-eic.github.io/doxygen/d4/d73/G4SPSEneDistribution_8cc_source.html

Using IAEA PSF format

Using IAEA PSFs

You need to:

Download the IAEA routines

https://www-nds.iaea.org/phsp/software/iaea_phsp_Sept2013.zip

Download the Geant4 class

https://www-nds.iaea.org/phsp/Geant4/

We've already provides these files in your program!

IAEA files you need to include in your program

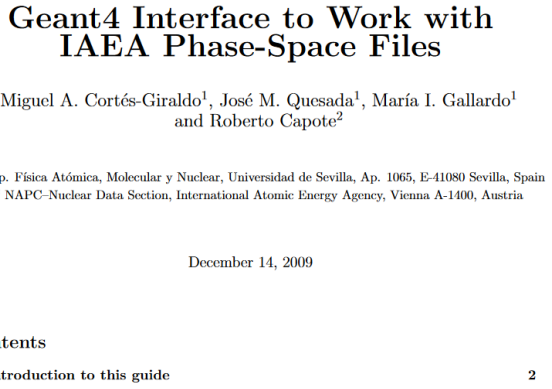
```
iaea_config.h
iaea_header.h/.cc
iaea_phsp.h/.cc
iaea_record.h/.cc
utilities.h/.cc
```

Writing a PSF in the IAEA format

Refer to the manual https://www-nds.iaea.org/phsp/Geant4/G4IAEAphsp_HowTo.pdf

One limitation of the IAEA format is that the particle types which it currently supports is: electrons, photons, positrons, neutrons and protons

See extra slides to see how to score your own PSF from scratch-where you control the power (any particle)



Geant4 Interface to Work with IAEA Phase-Space Files

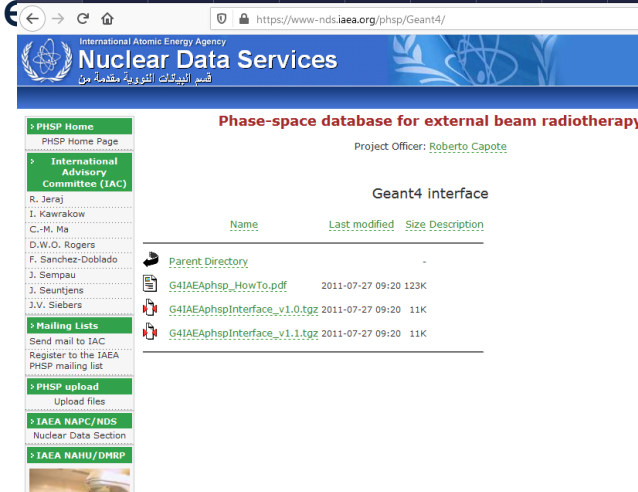
Miguel A. Cortés-Giraldo[1], José M. Quesada[1], María I. Gallardo[1] and Roberto Capote[2]

[1] Dep. Física Atómica, Molecular y Nuclear, Universidad de Sevilla, Ap. 1065, E-41080 Sevilla, Spain
[2] NAPC–Nuclear Data Section, International Atomic Energy Agency, Vienna A-1400, Austria

December 14, 2009

Contents

1 Introduction to this guide 2



If you wanted to do it yourself…

The Geant4 IAEA class was written before version 10 and requires some changes to compile. These are:

Add **G4SystemOfUnits.hh** to G4IAEA*.cc files to removes errors like:

```
#include "G4SystemOfUnits.hh"
```

```
/home/dave/G4school2019/ReadPSF/src/G4IAEAphspWriter.cc:297:51: error: ‘MeV’ was not declared in this scope
kinEnergyMeV = static_cast<IAEA_Float>(preE/MeV);
```

For two errors about exceptions in the **G4IAEAphspReader.hh:**

```
/home/dave/G4school2019/ReadPSF/include/G4IAEAphspReader.hh:87:63: error: no matching function for call to
‘G4Exception(const char [45])’
   { G4Exception("Cannot use G4IAEAphspReader void constructor"); }
```

Replace: `G4Exception("Error in G4IAEAphspReader::SetParallelRun()");`

With: `G4Exception("Error in G4IAEAphspReader::SetParallelRun()", "", JustWarning, "");`

Changes to your PrimaryGeneratorAction

//Contents of PrimaryGeneratorAction.hh

```
#ifndef PrimaryGeneratorAction_hh
#define PrimaryGeneratorAction_hh 1

#include "G4VUserPrimaryGeneratorAction.hh"
#include "G4Event.hh"
#include "globals.hh"

class G4Event;
class G4IAEAphspReader;

class PrimaryGeneratorAction : public
G4VUserPrimaryGeneratorAction
{
public:
PrimaryGeneratorAction();
~PrimaryGeneratorAction();
void GeneratePrimaries(G4Event*);

private:
  // Phase space reader
  G4IAEAphspReader* theIAEAReader;
};
#endif
```

//Contents of PrimaryGeneratorAction.cc

```
#include "PrimaryGeneratorAction.hh"
#include “OTHER.hh"
#include "G4IAEAphspReader.hh"

PrimaryGeneratorAction::PrimaryGeneratorAction()
{
//For IAEA Reader
G4String fileName = "FILENAME";
theIAEAReader = new G4IAEAphspReader(fileName);
}

PrimaryGeneratorAction::~PrimaryGeneratorAction()
{if (theIAEAReader) { delete theIAEAReader; }
}

void PrimaryGeneratorAction::GeneratePrimaries(G4Event* anEvent)
{
  theIAEAReader->GeneratePrimaryVertex(anEvent);
}
```

The "FILENAME"

To read in an IAEA PSF the “header” is also needed which provides details about the phase space file itself, such as number of entries and types of entries

The header and PSF must have the same name but different extensions eg: `CyberKnife_IRIS_5mm.IAEAphsp`

`CyberKnife_IRIS_5mm.IAEAheader`

So the filename you give is just the common part: eg: `CyberKnife_IRIS_5mm`

```
$ORIG_HISTORIES:
10000000
$PARTICLES:
3993789
$PHOTONS:
3993545
$ELECTRONS:
243
$POSITRONS:
1
```

- Let's download the CyberKnife_IRIS_5mm.IAEAphsp PSF (because it's small (only 110 MB versus GB+)
- To download the file to your current directory with terminal (you can just do it normally with the browser and GUI)

```
wget "https://www-nds.iaea.org/phsp/photon/CyberKnife_IRIS/CyberKnife_IRIS_5mm.IAEAphsp"
```

```
[dave@centaur2 ReadPSF]$ wget "https://www-nds.iaea.org/phsp/photon/CyberKnife_IRIS/CyberKnife_IRIS_5mm.IAEAphsp"
--2019-11-27 09:16:51--  https://www-nds.iaea.org/phsp/photon/CyberKnife_IRIS/CyberKnife_IRIS_5mm.IAEAphsp
Resolving www-nds.iaea.org (www-nds.iaea.org)... 104.20.23.134, 104.20.22.134, 2606:4700:10::6814:1786, ...
Connecting to www-nds.iaea.org (www-nds.iaea.org)|104.20.23.134|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 115819881 (110M)
Saving to: 'CyberKnife_IRIS_5mm.IAEAphsp'
```

Example Output-Notice anything strange?

Shooting one particle we get multiple tracks with a Parent ID = 0?

The IAEA format is setup to be very normalisation friendly

When Geant4 asks for 1 particle to be fired this corresponds with loading an event from the original simulation which generated the PFS. One event may have had many particles generated or no particles generated

So firing generating one event (beamOn 1) may result in many “primary” particles (parentID = 0) but have track IDs corresponding to the original simulation

On the flip side, you may fire 100 particles and when loading 100 events there may be no particle

```
**********************************************************************************************************
* G4Track Information:   Particle = gamma,   Track ID = 5,   Parent ID = 0
**********************************************************************************************************

Step#    X(mm)     Y(mm)     Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
    0   -0.302      1.01      -400     0.651        0        0         0 physicalWorld initStep
    1    -1.39      1.39      -150     0.651        0      250       250 physicalPhantom Transportation
    2    -1.61      1.47      -100     0.651        0       50       300   physScore Transportation
    3    -1.61      1.47      -100     0.651        0    0.001       300 physicalPhantom Transportation
    4    -1.81      1.53     -55.1     0.598 2.92e-05     44.9       345 physicalPhantom compt
    5      -27       -19      26.8     0.578 2.92e-05     88.1       433 physicalPhantom compt
    6      -55     -48.7      82.2     0.424 1.42e-05     68.9       502 physicalPhantom compt
    7     -150     -83.9       103     0.424        0      103       605 physicalWorld Transportation
    8   -1e+03      -399       292     0.424        0      926  1.53e+03  OutOfWorld Transportation

**********************************************************************************************************
* G4Track Information:   Particle = e-,   Track ID = 8,   Parent ID = 5
**********************************************************************************************************

Step#    X(mm)     Y(mm)     Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
    0      -55     -48.7      82.2     0.154        0        0         0 physicalPhantom initStep
    1      -55     -48.7      82.2     0.147  0.00625   0.0234    0.0234 physicalPhantom msc
    2    -54.9     -48.7      82.3     0.129   0.0187   0.0703    0.0937 physicalPhantom eIoni
    3    -54.9     -48.7      82.3     0.115   0.0137   0.0591     0.153 physicalPhantom eIoni
    4    -54.9     -48.7      82.3     0.091   0.0241   0.0516     0.204 physicalPhantom eIoni
    5    -54.8     -48.7      82.4    0.0629   0.0281   0.0395     0.244 physicalPhantom eIoni
    6    -54.8     -48.7      82.4    0.0487   0.0142   0.0274     0.271 physicalPhantom eIoni
    7    -54.8     -48.7      82.4    0.0303   0.0184   0.0221     0.294 physicalPhantom eIoni
    8    -54.8     -48.7      82.4    0.0162   0.0141   0.0149     0.308 physicalPhantom eIoni
    9    -54.8     -48.7      82.4         0   0.0162  0.00578     0.314 physicalPhantom eIoni

**********************************************************************************************************
* G4Track Information:   Particle = e-,   Track ID = 7,   Parent ID = 5
**********************************************************************************************************

Step#    X(mm)     Y(mm)     Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
    0      -27       -19      26.8    0.0198        0        0         0 physicalPhantom initStep
    1      -27       -19      26.8    0.0196 0.000249 0.000663  0.000663 physicalPhantom msc
    2      -27       -19      26.8         0   0.0196  0.00811   0.00877 physicalPhantom eIoni

**********************************************************************************************************
* G4Track Information:   Particle = e-,   Track ID = 6,   Parent ID = 5
**********************************************************************************************************

Step#    X(mm)     Y(mm)     Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
    0    -1.81      1.53     -55.1    0.0527        0        0         0 physicalPhantom initStep
    1     -1.8      1.54     -55.1    0.0521 0.000625  0.00373   0.00373 physicalPhantom msc
    2    -1.79      1.55     -55.1    0.0365   0.0156   0.0234    0.0271 physicalPhantom eIoni
    3     -1.8      1.56     -55.1     0.02   0.0166   0.0176    0.0447 physicalPhantom eIoni
    4    -1.79      1.56     -55.1         0     0.02  0.00837    0.0531 physicalPhantom eIoni

**********************************************************************************************************
* G4Track Information:   Particle = gamma,   Track ID = 4,   Parent ID = 0
**********************************************************************************************************

Step#    X(mm)     Y(mm)     Z(mm) KinE(MeV)  dE(MeV) StepLeng TrackLeng  NextVolume ProcName
    0    0.816   -0.0364      -400     0.297        0        0         0 physicalWorld initStep
    1    0.426    -0.313      -150     0.297        0      250       250 physicalPhantom Transportation
    2    0.348    -0.368      -100     0.297        0       50       300   physScore Transportation
    3    0.348    -0.368      -100     0.297        0    0.001       300 physicalPhantom Transportation
    4    0.334    -0.378     -90.8     0.219 2.83e-05     9.16       309 physicalPhantom compt
```

Creating/reading your own PSF

Summary of steps generating and using PSF

- Storing the PSF involves:
  - Set your desired beam configuration
  - Creating a “dummy volume” in the detector construction positioned in a “strategic location”
  - If a particle enters this volume, then the particles details will be stored in a file

- Reading in PSF involves:
  - Reading in PSF using the Primary Generator class

Generating the PSF in the Stepping Action

If a particle enters our plane we store its information and kill it

The PDGEncoding is a unique ID for every particle

Here we kill the track to:

Save time (not wasting resources to track it)

Prevent recording the same particle twice, for example if it scatters back in

```
//Retrieve particle information
G4String preVol  = aStep -> GetPreStepPoint() -> GetPhysicalVolume() ->
GetName();
G4String postVol = aStep -> GetPostStepPoint() -> GetPhysicalVolume() ->
GetName();

G4double KEpost = aStep -> GetPostStepPoint() -> GetKineticEnergy();
G4double pX = (aStep-> GetPostStepPoint() -> GetMomentumDirection().x());
G4double pY = (aStep-> GetPostStepPoint() -> GetMomentumDirection().y());
G4double pZ = (aStep-> GetPostStepPoint() -> GetMomentumDirection().z());

G4double postX = (aStep-> GetPostStepPoint() -> GetPosition().x());
G4double postY = (aStep-> GetPostStepPoint() -> GetPosition().y());
G4double postZ = (aStep-> GetPostStepPoint() -> GetPosition().z());

//-------------------PSF file generation--------------------
//----------Uncomment to generate phase space file----------
if (preVol != "physPSF" && postVol == "physPSF")
{
    G4ParticleDefinition *def = aStep->GetTrack()->GetDefinition();
   //Store particle information to root file
   analysis->FillPhaseSpace(def->GetPDGEncoding(), parentID,
postX/mm, postY/mm, pX, pY, pZ, KEpost/MeV);
   //kill track
   G4Track* theTrack = aStep->GetTrack();
   theTrack -> SetTrackStatus(fKillTrackAndSecondaries);
}
```

Saving the PSF in the Analysis Manager

We store the information with a Tree

```
//Filling function declaration in the header (.hh)
void FillPhaseSpace(Long64_t , Long64_t, Float_t ,
Float_t , Float_t , Float_t, Float_t , Float_t );
```

```
 //Tree's branches (PSF variables) in the
header
Long64_t PSpartID, PSparentID;
Float_t PSposX, PSposY, PSposZ, PSpx, PSpy,
PSpz, PSKE;
  TTree* result;
```

```
//Creating tree in source  (.cc)
result = new TTree("result", "test");

result->Branch("PSpartID", &PSpartID, "PSpartID/L");
result->Branch("PSparentID", &PSparentID, "PSparentID/L");
result->Branch("PSposY", &PSposY, "PSposY/F");
result->Branch("PSposZ", &PSposZ, "PSposZ/F");
result->Branch("PSpx", &PSpx, "PSpx/F");
result->Branch("PSpy", &PSpy, "PSpy/F");
result->Branch("PSpz", &PSpz, "PSpz/F");
result->Branch("PSKE", &PSKE, "PSKE/F");
```

```
//Fill function in source  (.cc)
void AnalysisManager::FillPhaseSpace(Long64_t tPSpartID,
Long64_t tPSparentID, Float_t tPSposY, Float_t tPSposZ,
Float_t tPSpx,  Float_t tPSpy, Float_t tPSpz, Float_t tPSKE)
{
PSpartID = tPSpartID;
PSparentID = tPSparentID;
PSposY = tPSposY; PSposZ = tPSposZ;
PSpx = tPSpx; PSpy = tPSpy;
PSpz = tPSpz; PSKE = tPSKE;
result->Fill();
}
```

Reading the PSF in the Primary Generator

```
void PrimaryGeneratorAction::ReadInPSFentries()
{
TFile f(PSFfileName.c_str(), "READ");
AnalysisManager* analysis = AnalysisManager::getInstance();

n = (TTree*)f.Get("ntuple1");
Long64_t PSpartID, PSparentID;
Float_t PSposX, PSposY, PSposZ, PSpx, PSpy, PSpz, PSKE;

//n->SetBranchAddress("ID",&particleType);

n->SetBranchAddress("PSpartID", &PSpartID);
n->SetBranchAddress("PSparentID", &PSparentID);
n->SetBranchAddress("PSposY", &PSposY);
n->SetBranchAddress("PSposZ", &PSposZ);
n->SetBranchAddress("PSpx", &PSpx);
n->SetBranchAddress("PSpy", &PSpy);
n->SetBranchAddress("PSpz", &PSpz);
n->SetBranchAddress("PSKE", &PSKE);
numberEntries = n->GetEntries();
//G4cout << numberEntries << G4endl;

rPartIDv.clear();
rxPosv.clear();
ryPosv.clear();
rzPosv.clear();
rpxv.clear();
rpyv.clear();
rpzv.clear();
rKEv.clear();
```

```
G4int randEntryNum = CLHEP::RandFlat::shoot(numberEntries -
entriesPerRead);

for (int ri = randEntryNum; ri < (randEntryNum + entriesPerRead);
ri++)//for random reading
{
n->GetEntry(ri);
rPartIDv.push_back(PSpartID);
//rAAv.push_back(rAA);
//rZZv.push_back(rZZ);
rxPosv.push_back(PSparentID);
ryPosv.push_back(PSposY);
rzPosv.push_back(PSposZ);
rpxv.push_back(PSpx);
rpyv.push_back(PSpy);
rpzv.push_back(PSpz);
rKEv.push_back(PSKE);
evtsRead++;

analysis->FillPrimXpos(PSposY);
analysis->FillPrimYpos(PSposZ);
}

//G4cout << "Stored entries. Closing root file." << G4endl;
f.Close();
eID = 0;
}
```

Typical User Primary Generator Action

//Contents of PrimaryGeneratorAction.cc

```
#include "PrimaryGeneratorAction.hh"
#include "G4ParticleGun.hh"
#include "G4ParticleTable.hh"
#include "G4Event.hh"
#include "G4GeneralParticleSource.hh"
//Various other headers

PrimaryGeneratorAction::PrimaryGeneratorAction()
{
//For particle gun (defining the beam here (in the PG class)
gun = new G4ParticleGun();
//GPSgun option
//GPSgun = new G4GeneralParticleSource();
}
PrimaryGeneratorAction::~PrimaryGeneratorAction()
{
delete gun;
//delete GPSgun;
}

void PrimaryGeneratorAction::GeneratePrimaries(G4Event* anEvent)
{
//SET primary beam characteristics
//Eg. Energy, position, momentum, particle type, shape
gun->GeneratePrimaryVertex(anEvent);

//If using GPS gun define characteristics in macro
//GPSgun->GeneratePrimaryVertex(anEvent);
}
```

//Contents of PrimaryGeneratorAction.hh

```
#ifndef PrimaryGeneratorAction_hh
#define PrimaryGeneratorAction_hh 1

#include "G4VUserPrimaryGeneratorAction.hh"
#include "G4Event.hh"
#include "globals.hh"

class G4GeneralParticleSource;
class G4ParticleGun;
class G4Event;

class PrimaryGeneratorAction : public
G4VUserPrimaryGeneratorAction {

public:
PrimaryGeneratorAction();
~PrimaryGeneratorAction();
void GeneratePrimaries(G4Event*);

private:
G4GeneralParticleSource* GPSgun;
G4ParticleGun* gun;
};

#endif
```

Calling a typical Primary Generator Action class

Calling/creating Primary Generator class in the "main" or "User Action Initialisation" class

```
 // Initialize the primary particles
PrimaryGeneratorAction* primary = new
PrimaryGeneratorAction();
SetUserAction(primary);
```

Typical structure of Primary Generator Action class

```
#include "PrimaryGeneratorAction.hh"
#include "G4Event.hh"
#include "G4ParticleGun.hh"
#include "G4GeneralParticleSource.hh"

PrimaryGeneratorAction::PrimaryGeneratorAction()
{
//For using the Particle Gun
gun  = new G4ParticleGun();
//For general particle source (GPS)
GPSgun  = new G4GeneralParticleSource();
}

PrimaryGeneratorAction::~PrimaryGeneratorAction()
{
delete gun;
delete GPSgun;
}
void PrimaryGeneratorAction::GeneratePrimaries(G4Event*
anEvent)
{...}
```