

Генерация первичных событий проект Detector_Parameterisation

Detector_Parameterisation.cc

DataWriter.cc (DataWriter.hh)

Loader.cc (Loader.hh)

Geometry.cc (Geometry.hh)

Action.cc (Action.hh)

PrimaryPart.cc (PrimaryPart.hh)

RunAct.cc (RunAct.hh)

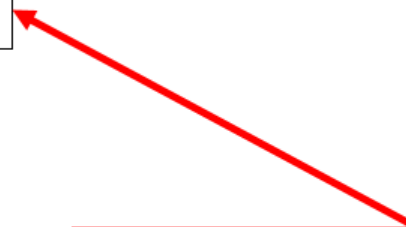
EventAct.cc (EventAct.hh)

StepAct.cc (StepAct.hh)

Si_det_Parameterisation.cc
(Si_det_Parameterisation.hh)



Генератор первичной вершины



Генерация первичных событий

Обязательным для реализации в проекте моделирования является класс G4VUserPrimaryGeneratorAction

Public Member Functions

	G4VUserPrimaryGeneratorAction ()
virtual	~G4VUserPrimaryGeneratorAction ()
virtual void	GeneratePrimaries (G4Event *anEvent)=0



Чисто виртуальный метод

PrimaryPart.hh

```
class PrimaryPart: public G4VUserPrimaryGeneratorAction
{
private:
    G4ParticleGun* GProton;
    G4ParticleGun* GNeutron;

public:
    PrimaryPart(std::ofstream&);
    ~PrimaryPart();

    std::ofstream *f_prim;
    virtual void GeneratePrimaries(G4Event* anEvent);
    G4ParticleGun* GetParticleGun() {return GProton;}
};
```

Класс PrimaryPart:

- ✓ наследник G4VUserPrimaryGeneratorAction;
- ✓ содержит объект-наследник G4ParticleGun класса G4VPrimaryGenerator;
- ✓ содержит описание метода GeneratePrimaries, в котором вызывается метод G4VPrimaryGenerator::GeneratePrimaryVertex(), создающий первичную вершину;
- ✓ в одном событии вершина может создаваться при участии нескольких объектов-наследников G4VPrimaryGenerator.

Генерация первичных событий

G4ParticleGun

- Производит первичную вершину из одной или нескольких частиц с заданными импульсом и начальным положением в пространстве
- Может управляться интерактивно
- Методы:

void SetParticleDefinition(G4ParticleDefinition*)

void SetParticleMomentum(G4ParticleMomentum)

void SetParticleMomentumDirection(G4ThreeVector)

void SetParticleEnergy(G4double)

void SetParticleTime(G4double)

void SetParticlePosition(G4ThreeVector)

void SetParticlePolarization(G4ThreeVector)

void SetNumberOfParticles(G4int)

Генерация первичных событий

PrimaryPart.cc

```
PrimaryPart::PrimaryPart(std::ofstream& ofsa)
{
    GProton = new G4ParticleGun(1);
    GProton->SetParticleDefinition(G4Proton::ProtonDefinition());
    GProton->SetParticleEnergy(100. * MeV);
    this->f_prim=&ofsa;
    (*f_prim) << std::setw(12) << "Hi from PrimaryPart!" << std::endl;
}
PrimaryPart::~~PrimaryPart()
{
    (*f_prim) << std::setw(12) << "Bye from PrimaryPart!" << std::endl;
}
PrimaryPart::GeneratePrimaries(G4Event* anEvent)
{
    GProton->SetParticlePosition(G4ThreeVector(0*mm , 0*mm, -5.*cm));
    GProton->SetParticleMomentumDirection(G4ThreeVector(0, 0, 1));
    GProton->GeneratePrimaryVertex(anEvent);
}
```

Генерация первичных событий

моноэнергетический точечный изотропный гамма - источник

```
gun = new G4ParticleGun(1);  
gun->SetParticleDefinition(G4Gamma::GammaDefinition());  
gun->SetParticlePosition(G4ThreeVector(0, 0, 0));  
gun->SetParticleEnergy(661 * keV);  
gun->SetParticleMomentumDirection(G4ThreeVector(G4UniformRand(), G4UniformRand(), G4UniformRand()));
```

G4UniforRand() возвращает случайное дробное число от 0 до 1

Интерактивное управление генератором – команды /gun/

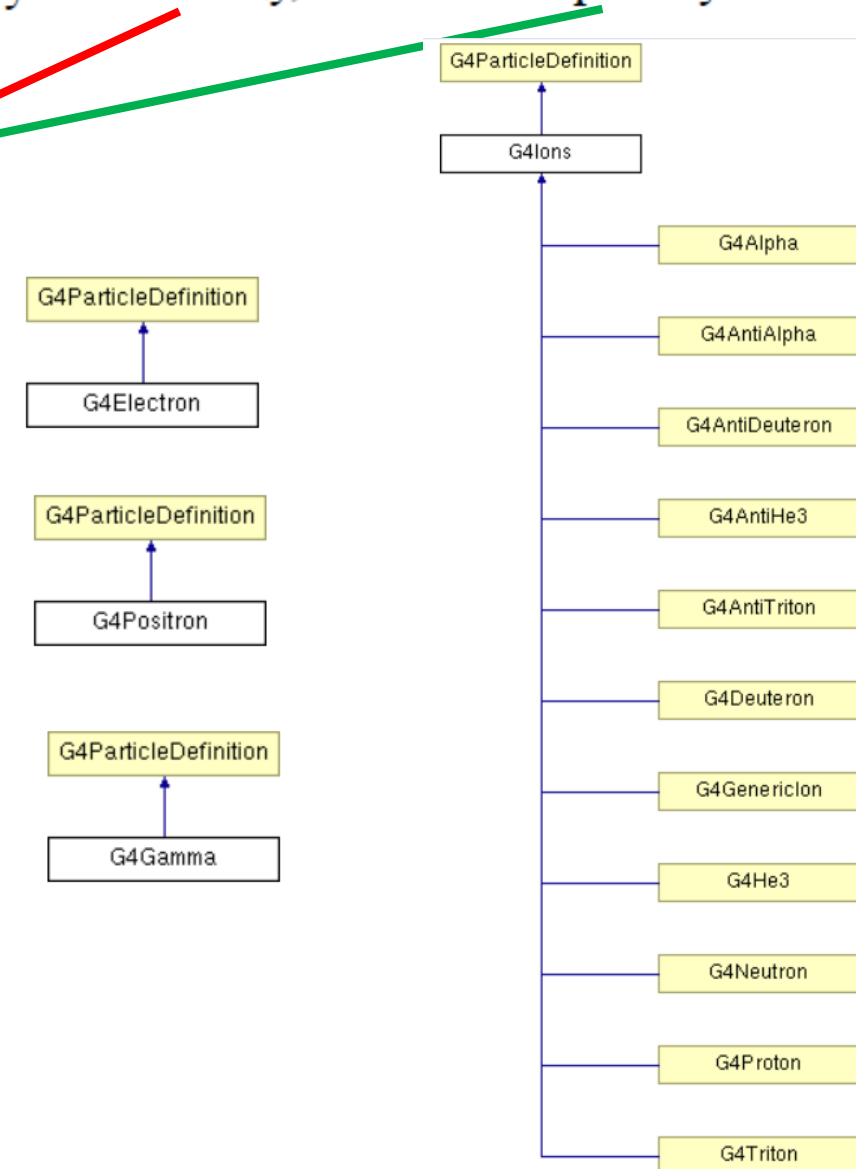
- /gun/List показать список частиц
- /gun/particle задать тип частицы
- /gun/direction установить направление вылета
- /gun/energy **установить кинетическую энергию**
- /gun/position установить координаты вершины
- /gun/time установить начальное время
- /gun/polarization задать поляризацию
- /gun/number задать число первичных частиц
- /gun/ion задать свойства иона

Частицы

G4ParticleGun позволяет одновременно настраивать как первичную частицу, так и её вершину.

Public Member 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)
G4ParticleDefinition *	GetParticleDefinition () const
G4ParticleMomentum	GetParticleMomentumDirection () const
G4double	GetParticleEnergy () const
G4double	GetParticleMomentum () const
G4double	GetParticleCharge () const
G4ThreeVector	GetParticlePolarization () const
G4int	GetNumberOfParticles () const



Частицы

Методы G4ParticleDefinition

G4String	GetParticleName()	название
G4double	GetPDGMass()	масса
G4double	GetPDGWidth()	ширина распада
G4double	GetPDGCharge()	заряд
G4double	GetPDGSpin()	спин
G4int	GetPDGiParity()	четность
G4int	GetPDGiConjugation()	зарядовое сопряжение
G4double	GetPDGIsospin()	изоспин
G4double	GetPDGIsospin3()	I_3
G4int	GetPDGiGParity()	G-четность
G4String	GetParticleType()	описание частицы
G4String	GetParticleSubType()	краткое описание частицы
G4int	GetLeptonNumber()	лептонный заряд
G4int	GetBaryonNumber()	барионный заряд
G4int	GetPDGEncoding()	код частицы согласно PDG
G4int	GetAntiPDGEncoding()	код соотв. античастицы

Основные коды PDG (Particle Data Group)

22 - гамма-квант
11 - электрон
-11 - позитрон
2212 - протон
2112 - нейтрон
+-100ZZZAAAI - ион
например
1000010020 - дейтрон
1000010030 - тритон
1000020040 - альфа
1000020030 - He3

Формирование PDG кода

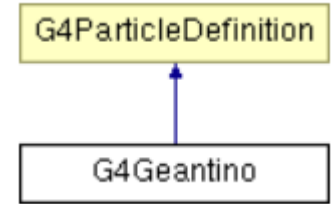
<https://pdg.lbl.gov/2007/reviews/montecarlohpp.pdf>

Категории частиц

- **Частицы, участвующие в трекинге**
 - стабильные частицы (протон, электрон, фотон ...)
 - долгоживущие ($>10^{-14}$ с) частицы (пион, мюон ...)
 - короткоживущие частицы, распад которых моделируется в Geant4 (π^0 ...)
 - К-мезоны
 - оптические фотоны
 - geantino
- **Ядра атомов**
 - легкие ядра (дейтрон, альфа-частица, ядро трития)
 - тяжелые ионы
- **Короткоживущие частицы**
 - кварки
 - глюоны
 - мезонные и барионные резонансы

в трекинге не участвуют
появляются только в некоторых
моделях физических процессов

geantino



Нейтральная частица, не имеющая физической природы, не участвующая в физических взаимодействиях, и используемая для отладки транспортировки через объёмы детектора.

Существует также заряженное geantino, которое кроме транспортирования через объёмы может взаимодействовать с электромагнитным полем.