**G4模板的使用说明**

Geant4的模拟程序的框架由CERN确定。不同的模拟程序的输入，输出，控制，相似，但不完全相同。为了避免重复的coding这部分，便是本模板的目的。

程序基于Geant4.10.01和Root 5.34两个库包。其中，Geant4安装了QT4，GDML的库；Root也选择了GDML的选项，也就是root可以读取GDML文件。

模拟程序主体是Geant4库包，但是，关于随机数的生成，一些数学函数，谱形（如能谱）的输入，模拟结果的存储，以及存储的结果的分析，都是基于Root库包的。所以想要使用本Geant4模板，Geant4+Root的基础知识都是必要的。

P.S.这里绝不建议为了快速入手Geant4，而避开Root的学习，避开Root中有用的库的使用，尤其是Random，Hitogram，TTree等。如果还不熟悉Root，这里建议先行学习Root（Root较为简单，可以作为C++的练兵场；Geant4和Root的配合可以在数据的存储，读取，处理上，省去很多精力）。

**认识配置文件**

程序的控制文件(\*\*\*.config)，放置在bin/Linux-g++/Config文件夹下。

**1，想要书写配置文件，首先应明白类Config的使用方法。**

所有的输入参数都存储在Config类中。Config使用静态map来存储配置信息。该类最好在程序的开始，就读入配置文件，在其他地方便可以直接调用。在本Template中，在主程序G4\_template.cc中调用。

{

Config\* configure = Config::Get\_Config\_Man();

}

初始化后，就可以使用Config定义好的函数来调用参数。这里我定义了六种函数来达到这个目的。

{

string Get\_ConfigValue\_string(string Paras);

int Get\_ConfigValue\_int(string Paras);

double Get\_ConfigValue\_double(string Paras);

bool Get\_ConfigValue\_stringArray(string Paras,int lower\_tem,int upper\_tem, string\* Value\_tem);

bool Get\_ConfigValue\_intArray(string Paras,int lower\_tem,int upper\_tem, int\* Value\_tem);

bool Get\_ConfigValue\_doubleArray(string Paras,int lower\_tem,int upper\_tem, double\* Value\_tem);

}

前三个函数是比较简单的，输入string类型的键，输出相应的Value。在写配置文件时，有些一个键对应多个Value，而且Value的类型可能不同，后面的三个函数就满足这个功能（第一个参数是键，第二个和第三个是你想要拿出来的Value的首尾标号，第四个参数用于存储取出来的Value）

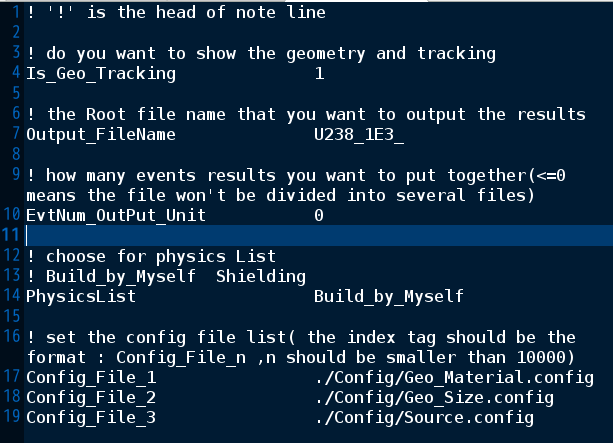
**关于这六个函数注意：**

1, 如果Config文件中没有相应的键，将会报错，同时返回来的值：对于string类型，返回“NotExisted”，对于int和double返回-999；

2, 对于读取数组的配置信息时，你要保证你提供的数组指针对应的数组有足够的空间用于存储，否则，将会出现“段错误”；

3, [lower\_tem,upper\_tem]对应的是lower\_tem<=Index<=upper\_tem。比如你想读键后面的三个数，lower\_tem=0, upper\_tem=2;

**2， 配置文件**

类Config对配置文件的扩展性较强。当你修改这个模板类时，可以根据自己的需要，添加配置文件，再利用上面的6个函数去读取这些配置信息。所以，现在在Config文件夹下的配置文件仅仅起到示范的作用，这里仅解释Initial.config文件。

图一，Initial.config文件

1, 配置文件中以“！” 或是“#”开头的行是注释行。

2, Line 4：是否显示径迹，1：显示，0：不显示；

3, Line 7：模拟数据存储文件的名字，程序会自动的在后面加上“\_0.root”，即：“Output\_FileName\_0.root”，如果第十行的EvtNum\_OutPut\_Unit不是<=0, 模拟过程中产生的新的文件将会依次为“Output\_FileName\_1.root”, ”Output\_FileName\_2.root”...

4, Line14：指定模拟过程中使用的Physicslist（一般使用Geant4预编译好的物理过程包就挺好的，比如Shielding，它就包括了所用的电磁过程以及低能中子反应）

5, Line17：这一行是为了扩展配置文件用的。当程序读到Config\_File\_n模式的键时，他会再去读取后面的配置文件。其中，1<=n<=10000（10000应该满足我们一般性质的需求！！！）。这些文件便是你发挥空间了。

**数据存储的思路**

Geant4是以step-by-step的方式进行模拟的，每一步中包含所用的模拟信息。一般我们不需要所用的模拟信息，如何提取我们关心的信息，便是这节要说明的。

这里使用的方法是绝大多数人都使用的方法（可能是唯一的方法）。

1，把我们关心的几何体设置为灵敏；

2，定义G4Hit，从G4Step中提取有用的信息，存放在这个类里；

3，在G4VSensitiveDetector中重载ProcessHits(G4Step\*)，在这个函数中，提取G4Step中的信息，将提取的信息赋给新创建的G4Hit，并将G4Hit存放在HitsCollection中；

4，在RunAction中定义ROOT File和TTree，并定义好相关的函数：这些函数可以给TTree的Branch赋值，并且可以TTree::Fill();

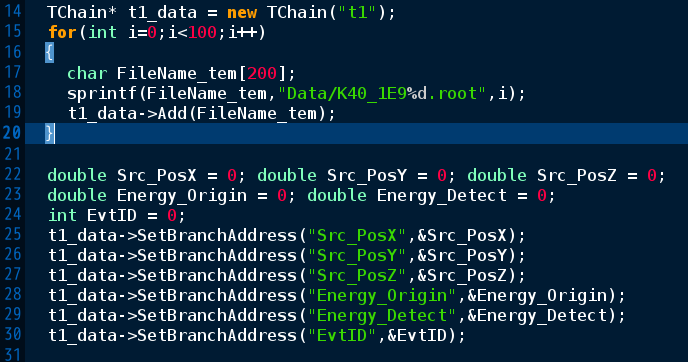
5，在EventAction中，利用第3步中生成的HitsCollection的信息，提取你关心的信息，并将这些信息赋给RunAction的Branch，然后Fill();

注意：一般，我将RunAction的指针赋给EventAction，将EventAction的指针再赋给PrimaryGeneratorAction，这样，PrimaryGeneratorAction中发射出来的粒子的信息可以通过EventAction传递给RunAction。因为，Geant4是模拟程序，我们很多时候是向Geant4学习，想知道自己的探测器对于不同的射线的响应是什么样子的。根据这个响应，我们可以改进探测器，也可以用于修正数据处理程序等等。

这种方法，我在这个Template中的一个应用是：在RunAction中定义一个能谱TH1D\* h1\_Src\_ESpec，打算使用这个直方图记录从particle gun中打出的粒子的能谱是不是我们期待的。通过EventAction，只要Particle gun中打出来一个粒子，不管EventAction是否记录这个事件，我都把这个能量Fill到这个直方图中。

**输出的模拟结果**

对于不同的模拟，我们关注的信息是不一样的。这个Template中，几何很简单，在world中放置一个NaI晶体，然后在原点处向他发射Gamma。这里我记录的信息是： 如果该Gamma在NaI有能量沉积，那么我记录发射出来的Gamma的位置，能量，以及在NaI中沉积的能量，以及该事件的一个编号。

下图是我用于分析数据时的TTree的结构。

图二，数据分析时，TTree的结构

当然生成的root文件中不仅仅存储了以TTree为结构的模拟数据，还存储了上面我们说到的particle gun的能谱h1\_Src\_ESpec。这个能谱仅仅用于校验，没有什么实质的作用。

**关于Detector的几何定义**

几何，材料的定义是Geant4中最基础的部分，也是最容易掌握的部分。但是有几个地方应该注意：

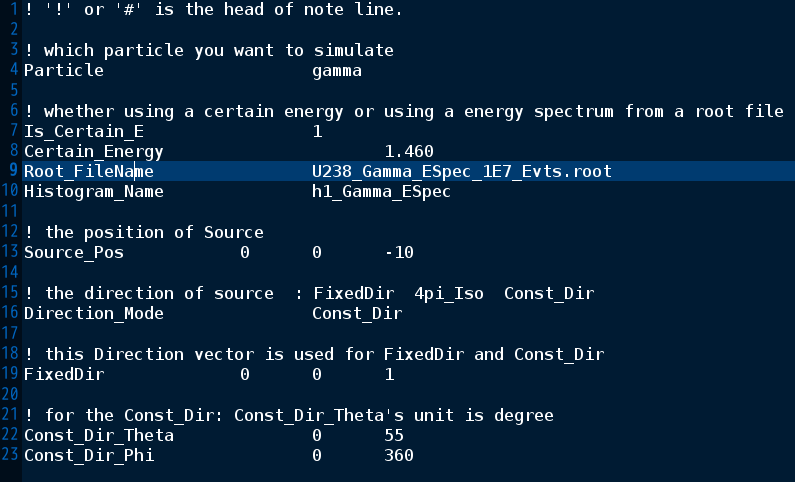
1，材料应该尽量使用Geant4中定义好的材料，这个可以参考G4说明书最后的部分；那里面的材料，每种元素都已经按照天然丰度计算的，简单而且准确；

2，在定义physical 几何体的时候，如果想要使用多个几何结构相同的探测器，或者是物理过程相同的探测器， Copy Number可以带来很大方便。他给每个physical探测器起个编号，这样，在EventAction中重建信号时，根据这个编号就知道，这个信息是那块探测器来的。

这种方法带来一个好处：不用定义多个SD类，从而不用定义多个HitsCollection。

**关于PrimaryGeneratorAction**

这里仅仅想说明一下与之相关的Source.config。



1，Line 4：Particle声明粒子的类型；

2，Line 7：Is\_Certain\_E说明发射出来的粒子是否是单一的能量；

3，Line 8：如果Is\_Certain\_E=1，那么键Certain\_Energy才有意义，否则这个键可有可无；

4，Line 9：如果Is\_Certain\_E=0，那么键Root\_FileName将会被使用，该文件用于存储Line10中的直方图；

5，Line13：发射出来的粒子的位置的三维坐标；

6，Line16：发射粒子的方向的模式，是4Pi同性的，是沿着一个方向的，是沿着一个方向还有个小张角的？如果是4Pi同性的那么后面的Line 19,22,23都没有意义。如果是沿着一个方向那么Line19指明了方向。如果是沿着一个方向张开的小角，那么Line19指明方向，Line22说明张开的角度是2\*55°，Line23说明Phi角的范围，0°是指从X轴开始，逆着Z轴看，逆时针转是正方向（就是我们通常的右手坐标系）。

7，这里长度单位是mm，能量单位是MeV，时间单位是ns；

**关于PhyiscsList**

到现在为止，我并没有在这块下很大的功夫，在仅仅考虑带电粒子的电离相互作用，或是Gamma的电磁相互作用时，直接使用例子ExampleN02的PhysicsList，在碰到中子或是重离子时，直接使用Shielding包。

在时间允许的情况下，应该开始这部分的学习，Geant4模拟的准确性主要取决于这部分。

**运行程序，需要注意两个地方：**

1, 首先运行脚本：. ./env.sh 将G4WORKDIR设置在当前目录下，这样编译生成的可执行文件就落在当前目录的./bin/Linux-g++；

2, 想要使用这个程序，当然不希望编译后的可执行文件的名字叫：G4\_template,这里需要修改两个地方：将主程序G4\_template.cc改为XXX.cc; 将GNUmakefile中的name := G4\_template改为name :=XXX,则make编译后的可执行文件就变成了XXX了。