Subject: Effective C++: Member initialisation
Posted by Volker Friese on Fri, 17 Feb 2012 08:14:33 GMT
View Forum Message <> Reply to Message

Hi

In the following I will explain how one get rid of one of the two most common warnings which show up when one switch on the effc++ warnings from gcc. The second warning is described in the
topic How to get rid of -Weffc++ warnings (part 2).

The first warning complaines about class members which are not initialized in the member initialization list. Please check example output below.

Quote:
warning: 'CbmFieldMap::fFileName' should be initialized in the member initialization list

Before we start with a real example from CBM code we have to do
some theory about C++ (taken from Scot Meyers Effective C++).

The problem is that one can't rely on that C++ will initialize Objects during declaration. C++ sometimes initialize the Objects during declaration, sometimes it doesn't do it.

int x;

In the above example x will be initialized (with 0) or not depending on the context. Due to this reason it is best practice to initialize the variables before usage. See example below.

int x = 0;
const char *text = "CBM is the best experiment ever."

When working with objects the initialization of all data members should be done in the constructor.

Coming back now to the CBM example. As example the class CbmFieldMap is taken.

The data members of this class are the following (taken from the header file)

```
TString  fFileName;
Double_t fScale;
Double_t fPosX, fPosY, fPosZ;
Double_t fXmin, fXmax, fXstep;
Double_t fYmin, fYmax, fYstep;
Double_t fZmin, fZmax, fZstep;
Int_t    fNx, fNy, fNz;
TArrayF* fBx;
TArrayF* fBy;
```

```
  TArrayF* fBz;
  Double_t fHa[2][2][2];
  Double_t fHb[2][2];
  Double_t fHc[2];
```

On of the constructors looks like

```
CbmFieldMap::CbmFieldMap()
{
  fPosX  = fPosY  = fPosZ  = 0.;
  fXmin  = fYmin  = fZmin  = 0.;
  fXmax  = fYmax  = fZmax  = 0.;
  fXstep = fYstep = fZstep = 0.;
  fNx    = fNy    = fNz    = 0;
  fScale = 1.;
  fBx    = fBy    = fBz    = NULL;
  fPosX = fPosY = fPosZ = 0.;
  fName     = "";
  fFileName = "";
  fType = 1;
}
```

To all the data memebers values are assigned in the body of the constructor. To mention here is the fact that one assignes values to fPosX, fPosY and fPosZ two times which shouldn't hurt but is definetly not something which is intended. The other point
to mention is the assignment of values to members of the base class. This is necessary because there is no appropriate constructor of the base class.

The problem here is that the data members have in the end the expected values but it is not the best solution for the task.
The C++ rules define that alle data members should be initialized before the body of the constructor is entered. In the above example the values of the data members were not initialized but assigned. The initialization take place when the default constructors are called before entering the functions body. This is not true for all the integrated types (int, float ...), because here it is not guaranteed that the data members are initalized at all before entering the function body.
There is much more in the reference (Effective C++), but as a rule of thumb you should remeber to initialize all data members in in the member initialization list. The order of the data members should be the same as the order they are declared in the header file.

The corrected constructur now looks like the one below. Before initializing the data members of the class the default constructor of the base class is "called". Then the data members show up in the same order as they are declared in the header file. In the body of the constructor only the default values to all elements of the arrays are assigned. I don't know any way to initalize the elements of an array in the member initalization list.
Also here the values of data members of the base class are
assigned because there is no appropriate constructpr of the base class. If there would be such a constructor one could call this constructor in the member initilization list.

```
CbmFieldMap::CbmFieldMap()
  : FairField(),
    fFileName(""),
    fScale(1.),
    fPosX(0.),
    fPosY(0.),
    fPosZ(0.),
    fXmin(0.),
    fXmax(0.),
    fXstep(0.),
    fYmin(0.),
    fYmax(0.),
    fYstep(0.),
    fZmin(0.),
    fZmax(0.),
    fZstep(0.),
    fNx(0),
    fNy(0),
    fNz(0),
    fBx(NULL),
    fBy(NULL),
    fBz(NULL)
{
  // Initilization of arrays is to my knowledge not
  // possible in member initalization lists
  for (Int_t i=0; i < 2 ; i++) {
    fHc[i]=0;
    for (Int_t j=0; j < 2 ; j++) {
      fHb[i][j]=0;
      for (Int_t k=0; k < 2 ; k++) {
  fHa[i][j][k]=0;
      }
    }
  }
  // Assign values to data members of base classes
  // There is no appropriate constructor of the base
  // class.
  fName = "";
  fType = 1;
}
```

I hope this topic helps to understand the warning and how to correct the code. Remarks and corrections are very welcome

Ciao

Florian