01/06/2011

Initialization

Hello world

dkfz.
GERMAN
CANCER RESEARCH CENTER
IN THE HELMHOLTZ ASSOCIATION

The birth of a variable

- Declaration
  - Known to compiler: name & type
  - Missing details
- Definition
  - Supply details missing above
  - Reserve memory, provide code body
- Initialization
  - Assign first value – usually via Constructor

But in C++ everything is initialized to 0

- NO
- Initialization to zero not guaranteed
  - Rule of thumb: If it is a C thing it is not initialized
    - Undeterministic behaviour

=> Make sure constructors initialize everything

First try

```
Class Birthdate{ … };
Class Person{
Public:
  Person(const string& name, const Birthdate& day);
Private:
  string theName;
  Birthdate theDay;
};

Person::Person(const string& name, const Birthdate& day)
{
theName = name;
theDay = day;
}
```

Are there any problems left?

First try – What did we (not) achieve?

- Guaranteed Initialization – no undetermined values
- BUT what we did were assignments not initializations
  - C++: Initialization takes place *before* the constructor body
    - Default Constructor was called and its work wasted

=> Use initialization list

Second try

```
Class Birthdate{ … };
Class Person{
Public:
  Person(const string& name, const Birthdate& day);
Private:
  string theName;
  Birthdate theDay;
};

Person::Person(const string& name, const Birthdate& day)
: theName(name),
theDay(day)
{
}
```

=> More efficient

More comments

- If only the default constructor is called it is done automatically, but:
  - Const and references can not be assigned and have to be initialized

- Order of initialization is defined by:
  - Base class before derived class
  - By declaration order
    - It will compile (most of the time) if initialized differently but don’t

Beware of Static

- The relative order of initialization of non-local static objects in different translation units is undefined

- There is no way you can make sure a non-local static object in another file is initialiazed before you access it in your own

- Can be solved by making the objects local

```
Object& AnObject()
{
Static Object thingy;
return thingy;
}
```

References and The End

- Meyers, Scott. *Effective C++: 55 Specific Ways to Improve Your Programs and Designs. ISBN 0-321-33487-6*

Questions?

Good luck hunting…