Subject: global Object and static std::map --> Crash
Posted by loki on Fri, 25 Sep 2009 23:38:11 GMT
View Forum Message <> Reply to Message

Hello,

I have a class with a static std::map member. If I create an object of this class in a function, then I have no problems. But if I create a global Object of this class, then the program crashes when operating on the std::map member in the constructor...
If I compile with MSC9, then it crashes at startup. If compiled with GCC it works, but not correct. Data is missing.

I made a testcase to verify, that its not my fault in an other place. The Testcase i attached at the bottom. (Testcase sometimes runs. But most times not)

I found out, that the initializationorder of global objects is not defined. But does this also effect the static member?
I mean, that first the object is created and then its static member??? At the moment it looks for me like this.

Also I tried a construct like this, but does not work too.

```
map<string, int>& GetMap()
{
   static map<string, int> theMap = map<string, int>;
   return theMap;
}
```

Maybe I am blind, but I cant see a mistake.
Please help me with this.

main.cpp

```
#include "Texture.h"
#include <iostream>

pb::gl::Texture tex("ABC");

int main()
{
 std::cout << "In main" << std::endl;

 return 0;
}
```

Texture.h

```
#ifndef _Texture_Texture_h_
#define _Texture_Texture_h_
```

```
#include <map>
#include <string>
#include <iostream>

namespace pb {
namespace gl {


class Texture
{
 public:
    Texture(std::string id);

 private:
  static std::map<std::string, int> textures;
};

}
}

#endif
```

Texture.cpp

```
#include "Texture.h"

using namespace pb::gl;
using namespace std;


map<string, int> Texture::textures = map<string, int>();

Texture::Texture(string id)
{
 map<string, int>::iterator element;
 element = textures.find(id); // <-- Crash inside of _tree
}
```

File Attachments

1) StaticMapCrash.zip, downloaded 587 times