Subject: Re: C++ templated class referencing each other
Posted by Xemuth on Sun, 11 Oct 2020 14:38:54 GMT
View Forum Message <> Reply to Message

Hello Lance,

Thanks for the time you take.

CompManagerEx.h

Did not worked for me. I had compilation error saying it was impossible to define/redefine CreateComponent because it was out of Upp (even if it was in Upp namespace)

Quote:Notice that I essentially put everything in the same header file (equivalently).
It dont work either, it fix the circulare dependencies between my problematic code and Object definition, however it also need the Scene object definition. If I include this definition in object.h file (since ComponentManager is in object.h file) it will lead to the same problem, circular dependencies include (because Scene include ObjectManager which include Object which will include Scene...)

Quote:Also, if your can guarantee that ComponentManager Will be the first member variable (because why-not and you should put a static_assert to avoid accidently add some other member variable in front it subsequently), you can probably save the
Object& obj;

member variable in the ComponentManager class. Simply define it like:
class Object;

```
class ComponentManager{
    public:
     ComponentManager(){}

     template <class T> T& CreateComponent(bool active, int position);

     /*
        Many templated function that force me to write declaration in this .h file
     */
    private:
      Object& object(){
          return *reinterpret_cast<Object*>(this);
      }

};
```

And change other part of your code accordingly.

I dont really understand this solution, are you meaning inherriting privatly/publicly Object from componentManager ? if yes I'm afraid the problem will remain since circulare reference between Scene and Object.h will appear. Maybe my architecture is bad and should be change to something like Opaque pointer

---