18 - CLASSES in C++

sexta-feira, 7 de fevereiro de 2025   07:01

- OOP is a styla on how to write your code
- C++ doesn't mipl certain impies but support it
- way to grudata and functionalites together
- Variables made of class are called object variables
    - And a new object is na instance of that class
- Defining a class we define the visibility of the variables and functions
    - By default the visibility is private, need to specif as public to acces or protected
- Fuctions inside classes are called methods
- USEFULL TO GROUP THINGS TOGETHER AND ADD FUNCTIONALITIES TO THE OBJECT

[CLASSES in C++](#)

CLASSES vs STRUCTS in C++

- Kind a similar one
- there is no much difference
- the main diference is the visibility options in structures (private, public, protected
    - Class is private by default
    - struct the default is public
- But this is tecnicly, but the use in code may differ
- struct exists by bacward compatibility with previous versions
    - the ompiler wouldn't know wht it was in old codes
- The usage differs
    - That is no right or wrong answer, differ by opinion
- struct used just to represent variables
- Never use a structure with inherence, go to classes

How to Write a C++ Class

- Log class to manage the log messages, used for debug process
- console is like na information dump
- Defined simple functions, member variables (public and private)
- Instantiated in main and also used the public functions

Static in C++

- 2 meanins,
    - outside of a class
        - Linkage of that symbel will be internal, only visible to that transation unit that you are working with (translation unit = file)
    - Inside of a class
        - All instances of that class will share the same memory, will only be one instance of that static variable across all instances of the class
- Focus on static outside of a class

Static for Classes and Structs in C++

- If used with a variable
    - Only one instance of taht variable across al isntances of that class
    - If one of the entity changes taht variable, it'll affect all other instances
    - Better to update the value by it's class than instance
        - By isntance could cause confusion and bugs
- Static method
    - Don't have access to the class instance
    - call without a class instance
    - canno write code that refer to a class instance

```cpp
struct StaticEntity22
{
    static int x22, y22;

    void Print(){
        std::cout << "Entity 22 x22 " << x22 << " Y22 " << y22 << std::endl;
    }
};
```

```cpp
94
95     StaticEntity22 se22;
96     se22.x22 = 2;
97     se22.y22 = 3;
98     se22.Print();
99
100    // StaticEntity22 se22_2 = {5, 8}; // This would fail for stati
101    StaticEntity22 se22_2; // This would fail for static classes
102    // se22_2.x22 = 5;
103    // se22_2.y22 = 8;
104    se22_2.Print(); // result should be the same as the other insta
105
106    // we can access them by the class and not by the instance
107    // And it'll change its value
108    StaticEntity22::x22 = 5;
109    StaticEntity22::y22 = 8;
110    StaticEntity22::Print(); // 5, 8
111
112    // Not static struct parameters
113    Entity22 e22;
114
115    e22.x22 = 2;
```

```cpp
    void Print(){
        std::cout << "Entity 22 x22 " << x22 << " Y22 " << y22 << std::endl;
    }
};

// When making this variables static, we need ot initialize
// them without any instantiation
int StaticEntity22::x22;
int StaticEntity22::y22;

int main()
{

    StaticEntity22 se22;
    e22.x22 = 2;
    e22.y22 = 3;
    e22.Print();

    // StaticEntity22 se22_2 = {5, 8}; // This would fail for static classes
    StaticEntity22 se22_2; // This would fail for static classes
    e22.x22 = 5;
    e22.y22 = 8;
    e22.Print();
```

```cpp
111
112     // Not static struct parameters
113     Entity22 e22;
114
115     e22.x22 = 2;
116     e22.y22 = 3;
117     e22.Print();
118
119     Entity22 e22_2 = { 5, 8 };
120
121     e22_2.Print();
122
```

PROBLEMS   OUTPUT   DEBUG CONSOLE   **TERMINAL**   PORTS

```
Hey
Hey
Hey

Hey
root@aee12d748e6b:/src/Dev/HelloWorld/out/build# ./HelloWorld
Static Entity 22 x22 2 Y22 3
Static Entity 22 x22 2 Y22 3
Static Entity 22 x22 5 Y22 8
Entity 22 x22 2 Y22 3
Entity 22 x22 5 Y22 8
```

Can access a non-static variable within a class, t generates na error

```cpp
struct StaticEntity22
{
    // static int x22, y22;
    int x22, y22;

    static void Print(){
        std::cout << "Static Entity 22 x22 " << x22 << " Y22 " << y22 << std::endl;
    }
};
```