40 - OPERATORS and OPERATOR OVERLOADING in C++

quinta-feira, 6 de março de 2025 07:29

- Operators
  - A symbol that we use inside of a function to peform something
  - dereference operator, equal…
  - we can give a new meaning to an operato, using an overload to change the behavior
    - changing a lot can leasd to bad programming
  - operators are basicly just functions
  - should be minimal and must make sense
  - If peaople needs to go to definition to understand what is going on, you failed
  -

```
struct Vector2_40 {
    float x, y;

    Vector2_40(float x, float y) : x(x), y(y) {};

    // marked as const because i'll not change anything here, just create a new obj
    Vector2_40 Add(Vector2_40 vec2) const {
        return Vector2_40(x + vec2.x, y + vec2.y);
    }

    Vector2_40 Multiply(Vector2_40 vec2) const {
        return Vector2_40(x * vec2.x, y * vec2.y);
    }

    // Define operators as any other functions
    Vector2_40 operator+ (Vector2_40 vec2) const {
        return Add(vec2);
    }

    Vector2_40 operator* (Vector2_40 vec2) const {
        return Multiply(vec2);
    }
};

// std::ostream& operator<<(std::string& stream, const Vector2_40& other){
//     stream << other.x << ", " << other.y;
//     return stream;
// }
```

  -

```
int main()
{
    Vector2_40 position40(4.0f, 2.0f);
    Vector2_40 speed40(4.0f, 2.0f);
    Vector2_40 Multiply40(4.0f, 2.0f);

    Vector2_40 result40 = position40.Add(speed40.Multiply(Multiply40));
    if(result40.x) std::cout << result40.x << std::endl;
    // THis is where it gets hard to read

    // let's use overloads
    Vector2_40 result40_2 = position40 + speed40 * Multiply40;
    if(result40.x) std::cout << result40_2.x << std::endl;

    // overload defined but not sure if it'll affect any other code
    // std::cout << result40_2 << std::endl;
```