Reinterpret Cast

در زبان C++ تقریباً همه چیز ممکن است! این کار هم به راحتی قابل انجام است ولی توصیه نمی شود! برای این کار ما مجبوریم هک کنیم!

```
#include <iostream>
using namespace std;
class Test
{
private:
	int x;
	char y;
public:
	Test() :x(0), y(0) { ; }
};

int main() {
	Test t;
	struct TestTwin
	{
		int tx;
		char ty;
	};
	reinterpret_cast<TestTwin*>(&t)->tx = 4;
	reinterpret_cast<TestTwin*>(&t)->ty = 'p';
	return 0;
}
```

علت و نحوه وقوع این اتفاق را میتوان با reinterpret_cast تحلیل کرد. reinterpret_cast نوعی cast کردن است که در C++ استفاده می شود، reinterpret_cast ویژگی های خاصی دارد که باعث شده است در این کار از آن بهره ببریم . ویژگی های زیر از خصوصیات آن است:

- بدون آنکه اهمیت دهد که کلاس ها به هم مرتبط هستند یا خیر، پوینتر ها را، بدون در نظر گرفتن تناسب، به هم تبدیل می کند.
- بررسی نمی کند که نوع و داده های اشاره گر توسط اشاره گر یکسان هستند یا خیر.

در سوال مد نظر ما یک استراکچر، که شبیه یا نوعی کلاس است، استفاده میکنیم و با reinterpret_cast پوینتری از آن استراکچر، که متوانیم مقدار دهی اش هم کنیم، -به صورت قاچاقی!- به آن عضو خصوصی کلاس مان مرتبط می کنیم! اینگونه است که در واقع عضو خصوصی کلاس مان را بدون آنکه فرند کنیم و یا از اینکپسولیشن استفاده کنیم، مقدار دهی کرده ایم!

این ویژگی تنها در زبان C++ برقرار است و زبان های برنامه نویسی دیگر این امکان را ندارند، چرا که هر شیء در سایر زبان های شئ گرا مانند جاوا اطلاعات مربوط به نوع زمان اجرا را با خود دارد. بنابراین وقتی شیء را به نوع دیگری تبدیل می کنیم ، اگر نوع جدید مطابقت نداشته باشد ، اینکار انجام نمی شود.

استفاده از این ویژگی، در واقع استفاده از ترفند های نه چندان زیبای پوینتر ها است که این کمی ساده و قابل درک نیست چراکه شما به نوعی غیر قانونی، عضو خصوصی کلاس خود را در بیرون از کلاس، بدون آنکه کلاس فرند باشد یا تابعی داشته باشید، مقدار دهی می کنید. علاوه بر آن استفاده مکرر از این ترفند، ما را با انبوهی از رفتار ها و اورهای غیر قابل پیشبینی روبرو میکند. بنا به دلایل ذکر شده این کار توصیه نمیشود. استفاده از access modifier ها ساده تر و قانونی تر! است و ریسک کمتر و معماری اصولی تری دارد، برای همین است که این کار و یا فرند کردن، نمیتواند جای استفاده از access modifier ها را بگیرند و ذره ای از ارزش های آنها کم کند!

امیرمهدی مختاری(۹۸۳۱۱۴۳)

Resources:

1. https://stackoverflow.com/questions/6717163/how-to-access-private-data-members-outside-the-class-without-making-friends
2. https://www.quora.com/How-do-we-access-private-data-members-of-a-class-outside-the-program-in-C++-language
3. https://www.geeksforgeeks.org/reinterpret_cast-in-c-type-casting-operators/#:~:text=reinterpret_cast%20is%20a%20type%20of,pointer%20is%20same%20or%20not.
4. https://stackoverflow.com/questions/573294/when-to-use-reinterpret-cast