Why are structure names different from their typedef names?

devblogs.microsoft.com/oldnewthing/20080326-00

March 26, 2008

Raymond Chen

In Windows header files, many structures are declared like this:

```
typedef struct tagXYZ {
 ...
} XYZ;
typedef struct _XYZ {
 ...
} XYZ;
/* there are other variations, too */
```

Why is the structure name different from typedef name?

This is a holdover from very early versions of the C language where structure tags, union tags, and typedefs were kept in the same namespace. Consequently, you couldn't say `typedef struct XYZ { ... } XYZ;`. At the open brace, the compiler registers `XYZ` as a structure tag name, and then when `XYZ` appears a second time, you get a redeclaration error. The standard workaround for this was to make the structure tag name a minor modification of the typedef name, most typically by putting the word `tag` in front.

The C language standardization process separated the structure and typename name spaces, so this workaround is no longer necessary, but it doesn't hurt either. Besides, even if new structures followed the `typedef struct XYZ { ... } XYZ;` pattern, you would just have people asking, "Why do some structures in `winuser.h` use the `tagXYZ` pattern and others use the `XYZ` pattern? Why can't it just be consistent?"

Next time, why you also don't see the pattern `typedef struct { ... } XYZ` very much either.

Raymond Chen

Follow