What are anonymous structs, and more importantly, how do I tell windows.h to stop using them?

devblogs.microsoft.com/oldnewthing/20170907-00

September 7, 2017

Raymond Chen

Windows header files take advantage of a language extension known as “anonymous structs” or “nameless structs”. It looks like this:

```
typedef struct _devicemode {
  TCHAR dmDeviceName[CCHDEVICENAME];
  WORD dmSpecVersion;
  WORD dmDriverVersion;
  WORD dmSize;
  WORD dmDriverExtra;
  DWORD dmFields;
  union {
    struct {
      short dmOrientation;
      short dmPaperSize;
      short dmPaperLength;
      short dmPaperWidth;
      short dmScale;
      short dmCopies;
      short dmDefaultSource;
      short dmPrintQuality;
    }; // <--- mystery #1
    struct {
      POINTL dmPosition;
      DWORD dmDisplayOrientation;
      DWORD dmDisplayFixedOutput;
    }; // <--- mystery #2
  }; // <--- mystery #3
  short dmColor;
  short dmDuplex;
  short dmYResolution;
  short dmTTOption;
  short dmCollate;
  TCHAR dmFormName[CCHFORMNAME];
  WORD dmLogPixels;
  DWORD dmBitsPerPel;
  DWORD dmPelsWidth;
  DWORD dmPelsHeight;
  union {
    DWORD dmDisplayFlags;
    DWORD dmNup;
  }; // <--- mystery #4
  DWORD dmDisplayFrequency;
#if(WINVER >= 0x0400)
  DWORD dmICMMethod;
  DWORD dmICMIntent;
  DWORD dmMediaType;
  DWORD dmDitherType;
  DWORD dmReserved1;
  DWORD dmReserved2;
#if (WINVER >= 0x0500) || (_WIN32_WINNT >= 0x0400)
  DWORD dmPanningWidth;
  DWORD dmPanningHeight;
#endif
#endif
} DEVMODE, *PDEVMODE, *LPDEVMODE;
```

Members of structures and unions normally have names. But in the `DEVMODE` structure, there are some members with no name. There's a union of two structures, and there's no name for the union (mystery #3); furthermore, the two structures that are members of the union also have no names (mysteries #1 and #2). And there's another union (mystery #4) that has no name.

Let's start with a smaller example. Consider this structure:

```
struct simple
{
 int a;
 union {
  int b;
  int c;
 } d;
} x;
```

In this example, we have a structure called `simple` and an instance of that structure in a variable called `x`. It consists of the following:

- An integer `a`, called `x.a`.
- An integer `b`, called `x.d.b`, which shares storage with
- An integer `c`, called `x.d.c`.

A *nameless union* omits the name `d`.

```
struct simple2
{
 int a;
 union {
  int b;
  int c;
 }; // <-- no name!
} x2;
```

This time, the contents are

- An integer `a`, called `x2.a`.
- An integer `b`, called `x2.b`, which shares storage with
- An integer `c`, called `x2.c`.

See what happened there? Omitting the name on the union means that the members of the union are accessible without having to say the name of the union (which is a good thing, because that union has no name).

Nameless unions are available in C and C++,[1] and that's what is happening in the `DEVMODE` structure. That solves mysteries #3 and #4.

These extensions are supported by both the Visual Studio compiler as well as the GCC compiler. But what if your compiler doesn’t?

The answer lies in the actual definition in the header file.

```
typedef struct _devicemodeW {
  WCHAR   dmDeviceName[CCHDEVICENAME];
  WORD   dmSpecVersion;
  WORD   dmDriverVersion;
  WORD   dmSize;
  WORD   dmDriverExtra;
  DWORD  dmFields;
  union {
    struct {
      short dmOrientation;
      short dmPaperSize;
      short dmPaperLength;
      short dmPaperWidth;
      short dmScale;
      short dmCopies;
      short dmDefaultSource;
      short dmPrintQuality;
    } DUMMYSTRUCTNAME; // magic #1
    struct {
      POINTL dmPosition;
      DWORD  dmDisplayOrientation;
      DWORD  dmDisplayFixedOutput;
    } DUMMYSTRUCTNAME2; // magic #2
  } DUMMYUNIONNAME; // magic #3
  short  dmColor;
  short  dmDuplex;
  short  dmYResolution;
  short  dmTTOption;
  short  dmCollate;
  WCHAR  dmFormName[CCHFORMNAME];
  WORD   dmLogPixels;
  DWORD  dmBitsPerPel;
  DWORD  dmPelsWidth;
  DWORD  dmPelsHeight;
  union {
    DWORD  dmDisplayFlags;
    DWORD  dmNup;
  } DUMMYUNIONNAME2; // magic #4
  DWORD  dmDisplayFrequency;
#if(WINVER >= 0x0400)
  DWORD  dmICMMethod;
  DWORD  dmICMIntent;
  DWORD  dmMediaType;
  DWORD  dmDitherType;
  DWORD  dmReserved1;
  DWORD  dmReserved2;
#if (WINVER >= 0x0500) || (_WIN32_WINNT >= 0x0400)
  DWORD  dmPanningWidth;
  DWORD  dmPanningHeight;
#endif
#endif
} DEVMODEW,*LPDEVMODEW,*PDEVMODEW;
```

There are magic symbols called `DUMMY` `SOMETHING` `NAME` where a name would normally go.

How curious.

If you then search the Windows header files for definitions of these magic symbols, you find them here in `winnt.h`:

```
//
// For compilers that don't support nameless unions/structs
//
#ifndef DUMMYUNIONNAME
#if defined(NONAMELESSUNION) || !defined(_MSC_EXTENSIONS)
#define DUMMYUNIONNAME   u
#define DUMMYUNIONNAME2  u2
#define DUMMYUNIONNAME3  u3
#define DUMMYUNIONNAME4  u4
#define DUMMYUNIONNAME5  u5
#define DUMMYUNIONNAME6  u6
#define DUMMYUNIONNAME7  u7
#define DUMMYUNIONNAME8  u8
#define DUMMYUNIONNAME9  u9
#else
#define DUMMYUNIONNAME
#define DUMMYUNIONNAME2
#define DUMMYUNIONNAME3
#define DUMMYUNIONNAME4
#define DUMMYUNIONNAME5
#define DUMMYUNIONNAME6
#define DUMMYUNIONNAME7
#define DUMMYUNIONNAME8
#define DUMMYUNIONNAME9
#endif
#endif // DUMMYUNIONNAME

#ifndef DUMMYSTRUCTNAME
#if defined(NONAMELESSUNION) || !defined(_MSC_EXTENSIONS)
#define DUMMYSTRUCTNAME  s
#define DUMMYSTRUCTNAME2 s2
#define DUMMYSTRUCTNAME3 s3
#define DUMMYSTRUCTNAME4 s4
#define DUMMYSTRUCTNAME5 s5
#else
#define DUMMYSTRUCTNAME
#define DUMMYSTRUCTNAME2
#define DUMMYSTRUCTNAME3
#define DUMMYSTRUCTNAME4
#define DUMMYSTRUCTNAME5
#endif
#endif // DUMMYSTRUCTNAME
```

Ah, now the pieces all fall into place.

If you define the symbol `NO NAMELESS UNION`, then the symbols `DUMMY SOMETHING NAME` are defined to expand to actual names. For dummy unions, they are `u`, `u2`, `u3`, and so on. For dummy structures, they follow the same pattern, but with `s` instead of `u`.

This means that if you indicate that you don't want the header files to use nameless unions, the nameless structures and unions magically get names! The names are not particularly exciting, but at least they have names.

```
DEVICEMODE dm;
dm.dmPosition = ...;       // if nameless unions are enabled
dm.u.s2.dmPosition = ...; // if nameless unions are disabled
```

Notice that I didn't use any Microsoft insider information to solve this mystery. All the information you need is right there, if you just follow the symbol definitions.

[1] The history here is unclear. Wikipedia claims that anonymous unions are in C++ and C11, but Stack Overflow claims that C++ supports anonymous unions only because C did. So there's some sort of circular causality loop here.

Raymond Chen

Follow