

[\(87 条消息\) union（联合体、结构体）的简介与使用_Xm 行墨的博客-CSDN 博客](#) [联合结构体](#)

union 叫共用体，又叫联合、联合体。“联合体”是一种特殊的类，也是一种构造类型的数据结构。在一个“联合体”内能够定义多种不同的数据类型。一个被说明为该“联合体”类型的变量中，同意装入该“联合体”所定义的不论什么一种数据。这些数据共享同一段内存，以达到节省空间的目的。

说了这么多，到底什么是联合体呢，就是在这个数据结构内，会有多种不同的数据，这些数据共同拥有同一段内存。。。好吧，可能感觉还不是很懂。看下面的代码片段，你可能就懂了。

```
1 typedef struct
2 {
3     unsigned char Red;
4     unsigned char Green;
5     unsigned char Blue;
6 }RGB_Typedef;
7
8 typedef union
9 {
10     RGB_Typedef rgb;
11     unsigned int value;
12 }Pix_Typedef;
```

这就声明了一个结构体和一个联合体，联合体内部包含了一个结构体和一个无符号整形数据（32位的）。刚刚我们说了联合体内部的数据共享同一段内存，意思就是说联合体内部的结构体的首地址和无符号整形数据的首地址是相同的，不信？你看下面的片段。

```
1 #include "stdio.h"
2
3 typedef struct
4 {
5     unsigned char Red;
6     unsigned char Green;
7     unsigned char Blue;
8 }RGB_Typedef;
9
10 typedef union
11 {
12     RGB_Typedef rgb;
13     unsigned int value;
14 }Pix_Typedef;
15
16 void main()
17 {
18     Pix_Typedef pix;
19     printf("%x\r\n",&pix.rgb);
20     printf("%x\r\n",&pix.value);
21 }
```

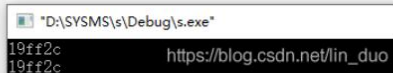
运行结果截图：

```
#include "stdio.h"

typedef struct
{
    unsigned char Red;
    unsigned char Green;
    unsigned char Blue;
}RGB_Typedef;

typedef union
{
    RGB_Typedef rgb;
    unsigned int value;
}Pix_Typedef;

void main()
{
    Pix_Typedef pix;
    printf("%x\r\n",&pix.rgb);
    printf("%x\r\n",&pix.value);
}
```



可以清晰的看到，联合体里面的结构体和无符号整形数据是相同的。

我们知道了联合体的功能了，那么到底会有一些怎么样的应用呢？

我们都知道RGB三原色是有红、绿、蓝各占一个字节（0-255）表示的，有些时候，我们需要单独去R、G、B三个色值，有些时候，又需要合在一起使用。难道我们每一次使用的时候，都对数据进行拆分、融合？这样实在是太麻烦了。我们就可以利用联合体的方式来定义三原色。

```
1 typedef struct
2 {
3     unsigned char Red;
4     unsigned char Green;
5     unsigned char Blue;
6 }RGB_Typedef;
7
8 typedef union
9 {
10     RGB_Typedef rgb;
11     unsigned int value;
12 }Pix_Typedef;
```

如果我们定义三原色分别为0X11、0X22、0X33，然后不再做任何处理，直接打印value。会是怎么一种情况。代码如下

```
1 #include "stdio.h"
2
3 typedef struct
4 {
5     unsigned char Red;
6     unsigned char Green;
7     unsigned char Blue;
8 }RGB_Typedef;
9
10 typedef union
11 {
12     RGB_Typedef rgb;
13     unsigned int value;
14 }Pix_Typedef;
15
16 void main()
17 {
18     Pix_Typedef pix;
19     pix.rgb.Blue=0X33;
20     pix.rgb.Green=0X22;
21     pix.rgb.Red=0X11;
22     printf("%X\r\n",pix.value);
23 }
```

复制

运行结果如下图:

```
#include "stdio.h"

typedef struct
{
    unsigned char Red;
    unsigned char Green;
    unsigned char Blue;
}RGB_Typedef;

typedef union
{
    RGB_Typedef rgb;
    unsigned int value;
}Pix_Typedef;

void main()
{
    Pix_Typedef pix;
    pix.rgb.Blue=0X33;
    pix.rgb.Green=0X22;
    pix.rgb.Red=0X11;
    printf("%X\r\n",pix.value);
}
```

"D:\SYSMS\s\Debug\s.exe"

CC332211

Press any key to continue

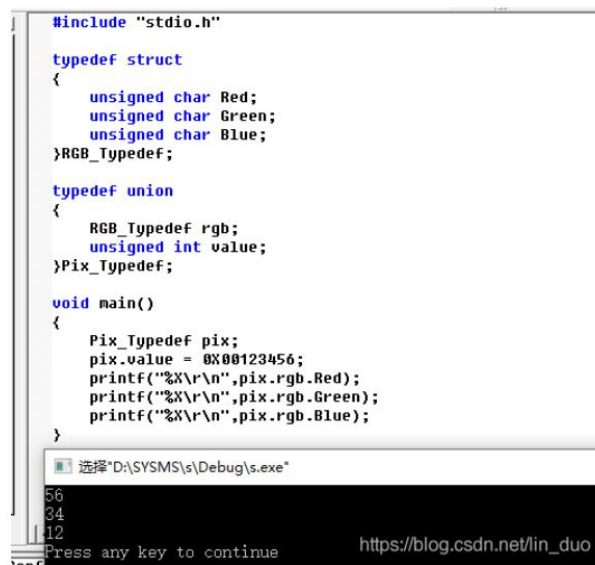
https://blog.csdn.net/lin_duo

这里需要说明一下的是, unsigned int 是一个4字节数据, 而前面我们只定义了三个unsigned char, 只占三个字节。所以MSB自动补齐了。在union中, 分配内存空间的大小, 等于占内存最大的数据类型字节大小。

在上面, 我们已经看到可以直接通过定义R、G、B值, 来修改融合后的值。当然, 反过来也是可以的。我们首先对value赋值, 然后再输出R、G、B值。代码如下:

```
1  #include "stdio.h"
2
3  typedef struct
4  {
5      unsigned char Red;
6      unsigned char Green;
7      unsigned char Blue;
8  }RGB_Typedef;
9
10 typedef union
11 {
12     RGB_Typedef rgb;
13     unsigned int value;
14 }Pix_Typedef;
15
16 void main()
17 {
18     Pix_Typedef pix;
19     pix.value = 0X00123456;
20     printf("%X\r\n",pix.rgb.Red);
21     printf("%X\r\n",pix.rgb.Green);
22     printf("%X\r\n",pix.rgb.Blue);
23 }
```

运行结果如下图：



```
#include "stdio.h"

typedef struct
{
    unsigned char Red;
    unsigned char Green;
    unsigned char Blue;
}RGB_Typedef;

typedef union
{
    RGB_Typedef rgb;
    unsigned int value;
}Pix_Typedef;

void main()
{
    Pix_Typedef pix;
    pix.value = 0X00123456;
    printf("%X\r\n",pix.rgb.Red);
    printf("%X\r\n",pix.rgb.Green);
    printf("%X\r\n",pix.rgb.Blue);
}
```

选择"D:\SYSMS\s\Debug\s.exe"

56
34
12

Press any key to continue https://blog.csdn.net/in_duo

由此，可以说明，无论是修改结构体，还是修改unsigned int，都会对对方造成影响。因为他们**是共享同一段内存**。