Linux 内核中#,##,_ _VA_ARGS_ _的用法

2011-11-13 21:03

连接符 ##(两个井号)

不知道什么符 #(一个井号)

连接符号由两个井号组成，其功能是在带参数的宏定义中将两个子串(token)联接起来，从而形成一个新的子串。但它不可以是第一个或者最后一个子串。所谓的子串(token)就是指编译器能够识别的最小语法单元。具体的定义在编译原理里有详尽的解释

#符是把传递过来的参数当成字符串进行替代。

假设程序中已经定义了这样一个带参数的宏：

```
#define PRINT( n ) printf( "token" #n " = %d", token##n )
```

同时又定义了二个整形变量：

```
int token9 = 9;
```

现在在主程序中以下面的方式调用这个宏：

```
PRINT( 9 );
```

那么在编译时，上面的这句话被扩展为：

```
printf( "token" "9" " = %d", token9 );
```

注意到在这个例子中，PRINT(9);中的这个”9”被原封不动的当成了一个字符串，与”token”连接在了一起，从而成为了 token9。而#n 也被”9”所替代。

可想而知，上面程序运行的结果就是在屏幕上打印出 token9=9

还有点不明白?

再来一个例子:

```
#define PRINT( n ) printf( "token" #n " = %d", game##n )

int token9 = 9;

int game9 = 99;
```

调用:

PRINT(9);

屏幕上打印出:

token9 = 99;

转自： http://thatax.blog.163.com/blog/static/208926802008823918271l6/

1.#

假如希望在字符串中包含宏参数，ANSI C 允许这样作，在类函数宏的替换部分，#符号用作一个预处理运算符，它可以把语言符号转化程字符串。例如，如果 x 是一个宏参量，那么#x 可以把参数名转化成相应的字符串。该过程称为字符串化（stringizing）.

```
#incldue <stdio.h>
#define PSQR(x) printf("the square of" #x "is %d./n",(x)*(x))
int main(void)
{
        int y =4;
        PSQR(y);
        PSQR(2+4);
        return 0;
}
```

输出结果：

```
the square of y is 16.
the square of 2+4 is 36.
```

第一次调用宏时使用“y”代替#x；第二次调用时用“2+4"代#x。

2.##

##运算符可以使用类函数宏的替换部分。另外，##还可以用于类对象宏的替换部分。这个运算符把两个语言符号组合成单个语言符号。例如：

```
#define XNAME(n) x##n
```

这样宏调用：

```
XNAME(4)
```

展开后：

```
x4
```

程序：

```
#include <stdio.h>
#define XNAME(n) x##n
#define PXN(n) printf("x"#n" = %d/n",x##n)
int main(void)
{
        int XNAME(1)=12; //int x1=12;
```

```
        PXN(1); //printf("x1 = %d/n", x1);
        return 0;
}
```

3.可变宏 ...和_ _VA_ARGS_ _

实现思想就是宏定义中参数列表的最后一个参数为省略号（也就是三个点）。这样预定义宏_ _VA_ARGS_ _就可以被用在替换部分中，以表示省略号代表什么。比如：

```
#define PR(...) printf(_ _VA_ARGS_ _)
PR("hello");-->printf("hello");
PR("weight = %d, shipping = $.2f",wt,sp); -->printf("weight = %d, shipping = $.2f",wt,sp);
```

省略号只能代替最后面的宏参数。

#define W(x,...,y)错误!