linux 调试 之printf - 小小步伐

小小步伐关注 - 2 粉丝 - 0 +加关注

1. 可变参数宏

1.1 C99 标准中可变参数宏

```
#define debug(format, ...) printf(format,
```

1.2 gcc 复杂宏

```
 #define debug(format, args...) printf(format,
```

#的作用 ： 连接两个宏,如果可变参数被忽略或为空,"##"操作将使预处理器(preprocessor)去除掉它前面的那个逗号.

2. 打印带颜色的字符串

2.1 格式

```
printf("\033[字背景颜色;字体颜色m字符串\033[0m" );
```

2.2 颜色码

```
背景色                          前景色
40: 黑                           30: 黑
41: 红                           31: 红
42: 绿                           32: 绿
43: 黄                           33: 黄
44: 蓝                           34: 蓝
45: 紫                           35: 紫
46: 深绿                        36: 深绿
47: 白色                        37: 白色
```

3. 举例----打印调试宏

```
#ifndef __USE_DEBUG
#define __USE_DEBUG

#ifdef  USE_DEBUG
#define DEBUG_LINE() printf("[%s:%s] line=%d\r\n",__FILE__, __func__, __LINE__)
#define DEBUG_ERR(fmt, args...) printf("\033[47;31m[%s:%d]\033[0m   "fmt" \r\n", __func__, __LINE__,  ##args)
#define DEBUG_INFO(fmt, args...) printf("\033[33m[%s:%d]\033[0m  "fmt"  \r\n", __func__, __LINE__, ##args)
#else
#define DEBUG_LINE()
#define DEBUG_ERR(fmt, ...)
#define DEBUG_INFO(fmt, ...)
#endif

#endif



int main(int args, const char ** argv)
{
    int fd;

    fd = open("/dev/nothing, O_RDWR");
    if(fd < 0){
        DEBUG_ERR(“open failed , ret = %d, %m\n”, fd);
    }
}
```

运行：

```
linux@fyang:~/work/debug$ gcc debug.c -DUSE_DEBUG
linux@fyang:~/work/debug$ ./a.out
[main:42] open faild, ret = -1, No such file or directory
linux@fyang:~/work/debug$
```

注： %m 会打印errno 对应的字符串 功能类似 perror