

唐佐林视频教程

狄泰未来

第22课

程序异常处理的设计（上）

© 2018 成都狄泰未来科技有限公司



程序异常处理的设计

- 在开发中，**不可避免的**需要进行异常处理
- 函数调用时的异常：
 - **并不是指函数设计上的逻辑错误**
 - **而是可预见的非正常功能的执行分支**



```
int is_digit(const char* s)
{
    int ret = 1;

    if( !s ) return -1; // 异常处理分支

    while( *s && (ret = ret && ('0' <= *s) && (*s <= '9')) )
        s++;

    return ret;
}
```

程序异常处理的设计

- 异常处理的意义
 - 软件开发过程中，大多数情况下都在处理异常情况
 - 异常不是错误，但是可能导致程序无法正常执行
 - 异常处理直接决定软件产品的鲁棒性和稳定性



程序异常处理的设计

- 项目中的异常设计
 - 实现表达异常的通用方法（异常如何表示？）
 - 设定异常报告的方法（发生了什么异常？如何知道？）
 - 制定统一处理异常的原则（怎么处理异常？）

程序异常处理的设计

- 在 C 语言中通过错误码对异常进行表示
 - 优势：
 - 错误码的定义简单，使用方便
 - 劣势：
 - 同一个错误码可能表示不同含义



程序异常处理的设计

- 异常表示的通用设计方法

采用整数分区域的方式对异常进行表示



```
#ifndef __error_t_defined
typedef int error_t;
#define __error_t_defined
#endif
```

```
#define MODULE_BITS 15
#define ERROR_BITS 16
```

注意：

一般而言，错误码最高位恒为 1，即：所有错误码为负数！

程序异常处理的设计

- 错误码类型的操作

- ERROR_MARK // 0x80000000 错误码最高位为 1
- ERROR_BEGIN(_module_id) // 根据模块 ID 计算起始错误号
- ERROR_T(_module_error) // 根据错误号生成错误码
- MODULE_ERROR(_error_t) // 获取错误码中的模块内错误号 (低 16 位)
- MODULE_ID(_error_t) // 获取错误码中的模块 ID (高 15 位)

The shortest answer is doing.

编程实验

错误码的设计与使用

demo1



程序异常处理的设计

- 异常的报告
 - 通常情况下，系统日志的是异常报告的主要方式
 - 注意：异常报告并不是异常处理
 - 异常报告用于记录异常的发生
 - 异常处理用于阻止异常导致的程序崩溃



程序异常处理的设计

异常报告与异常处理示例（整体框架）

```
#define HANDLER(err) do { \
    int e=err; \
    if((e & ERROR_MARK) && \
        !err_handler(e)) \
        exit(e); \
} while(0)
```

异常处理函数


```
int do_something()
{
    while(1)
    {
        HANDLER(sub_func());

        // HANDLER(sub_func2());

        // ...
    }

    return 0;
}
```

处理函数可能发生的异常



程序异常处理的设计

- 异常报告与异常处理示例（异常报告）

```
int sub_func()
{
    int ret = 0;

    // do some task

    if(ret & ERROR_MARK)
        printf("sub_func: %X\n", ret);

    // return 0 for no exception
    return ret;
}
```

程序异常处理的设计

- 异常报告与异常处理示例（异常处理）

```
int err_handler(int e)
{
    int ret = 0;

    if( e & ERROR_MARK )
        switch(e ^ ERROR_MARK)
        {
            // handle something unexcepted
        }
    else
        ret = 1;

    // return 1 for success handling
    return ret;
}
```

程序异常处理的设计

- 工程开发时的一些建议
 - 尽量在异常发生的地方报告异常
 - 有助于事后找到异常发生时的函数调用路径
 - 尽量在上层函数中统一处理异常
 - 集中处理异常有助于提高代码的维护性

Nothing seek, nothing find.

实例分析

异常报告与处理

demo2



小结

- C 项目中通常采用整数分区域的方式对异常进行表示
- 异常号包含了模块信息以及模块相关的具体异常信息
- 通常情况下，系统日志的是异常报告的主要方式
- 尽量就近报告异常，尽量统一处理异常



程序异常处理的设计

- To be continued ...

当前的设计中，直接输出异常号的方式易于
问题定位吗？是否更好的异常输出方式？



问与答

- Q / A

