

KOMyself的专栏

目录视图

摘要视图

RSS 订阅

个人资料



KOMyself

访问：115702次

积分：1339

等级：

BLOG

>4

排名：千里之外

原创：15篇

转载：54篇

译文：0篇

评论：4条

文章搜索

文章分类

Windows Phone (7)

Asp.net (2)

c# (7)

regular expression (1)

ios (43)

C++/C (1)

文章存档

2013年07月 (1)

2013年03月 (2)

2013年02月 (3)

2013年01月 (1)

2012年12月 (4)

展开

阅读排行

AVPlayer 多媒体播放器 (17050)

ios开发，将子线程获取的

深度学习代码专栏 攒课-我的学习我做主 开启你的知识管理，知识库个人图谱上线

#ifdef,#else,#endif,#if用法详解(转)

标签：[语言](#) [include](#) [class](#) [math.h](#) [扩展](#) [c](#)

2012-07-24 10:56 2644人阅读 评论(0) 收藏 举报

分类：[ios \(42\)](#)

预处理就是在进行编译的第一遍词法扫描和语法分析之前所作的工作。说白了，就是对源文件进行编译前，先对预处理部分进行处理，然后对处理后的代码进行编译。这样做的好处是，经过处理后的代码，将会变的很简短。

关于预处理命令中的文件包含（#include），宏定义（#define），书上已经有了详细的说明，在这里就不详述了。这里主要是对条件编译（#ifdef,#else,#endif,#if等）进行说明。以下分3种情况：

1: 情况1:

```
#ifdef _XXXX

...程序段1...

#else

...程序段2...

#endif
```

这表明如果标识符_XXXX已被#define命令定义过则对程序段1进行编译；否则对程序段2进行编译。

例：

```
#define NUM

.....

.....

.....

#ifdef NUM

printf("之前NUM有过定义啦! :) \n");

#else

printf("之前NUM没有过定义! :( \n");

#endif

}
```

如果程序开头有#define NUM这行，即NUM有定义，碰到下面#ifdef NUM的时候，当然执行第一个printf。否则第二个printf将被执行。

我认为，用这种，可以很方便的开启/关闭整个程序的某项特定功能。

2:情况2:

```
#ifndef _XXXX

...程序段1...

#else

...程序段2...
```

ios 通讯录“写”操作大全 (11835)
Objective-C 对 URL 进行 (9583)
ios切换View (6890)
ios-获取系统相簿里边的I (5678)
iphone/ipad关于size, frai (3592)
IOS Socket使用大全 (2864)
Invalid Cross-thread acc (2805)
#ifndef,#else,#endif,#if用法 (2732)
#ifndef,#else,#endif,#if用法 (2638)

评论排行

AVPlayer 多媒体播放器 (3)
JSON库解析json文件 (1)
WP7中怎样进行网络请求 (0)
非常棒的天气预报代码 (0)
好久没更新博客了 (0)
IIS上运行aspx发现出现朋 (0)
网上总结到的GridView自 (0)
TextBox的AutoPostBack (0)
Objective-C 对 URL 进行 (0)
学习了jquery实现滑动效 (0)

推荐文章

* 2016 年最受欢迎的编程语言是什么?
* Chromium扩展 (Extension) 的页面 (Page) 加载过程分析
* Android Studio 2.2 来啦
* 手把手教你做音乐播放器 (二) 技术原理与框架设计
* JVM 性能调优实战之: 使用阿里开源工具 TProfiler 在海量业务代码中精确定位性能代码

最新评论

JSON库解析json文件
qq_31150773: 为什么我导入进去, 报错呀, ,
AVPlayer 多媒体播放器
jh10622110: 转的也不放别人的原地址, 能有点素质么
AVPlayer 多媒体播放器
cwli: 亲, 有源码木的, 瞅一眼...
AVPlayer 多媒体播放器
Hunter_Tang: 能不能把源代码发一份给我? 遇到AVPlayer播放视频的问题。如果可以麻烦发致邮箱497658350...

#endif

这里使用了#ifndef, 表示的是if not def。当然是和#define相反的状况 (如果没有定义了标识符_XXXX, 那么执行程序段1, 否则执行程序段2)。

3: 情况3:

#if 常量

...程序段1...

#else

...程序段2...

#endif

这里表示, 如果常量为真 (非0, 随便什么数字, 只要不是0), 就执行程序段1, 否则执行程序段2。

我认为, 这种方法可以将测试代码加进来。当需要开启测试的时候, 只要将常量变1就好了。而不要测试的时候, 只要将常量变0。

ifdef #ifndef 等用法

文件中的#ifndef

头件中的#ifndef, 这是一个很关键的东西。比如你有两个C文件, 这两个C文件都include了同一个头文件。而编译时, 这两个C文件要一同编译成一个可运行文件, 于是问题来了, 大量的声明冲突。

还是把头文件的内容都放在#ifndef和#endif中吧。不管你的头文件会不会被多个文件引用, 你都要加上这个。一般格式是这样的:

#ifndef <标识>

#define <标识>

.....

.....

#endif

<标识>在理论上来说可以是自由命名的, 但每个头文件的这个“标识”都应该是唯一的。标识的命名规则一般是头文件名全大写, 前后加下划线, 并把文件名中的“.”也变成下划线, 如: stdio.h

#ifndef _STDIO_H_

#define _STDIO_H_

.....

#endif

2. 在#ifndef中定义变量出现的问题 (一般不定义在#ifndef中)。

#ifndef AAA

#define AAA

...

int i;

...

#endif

里面有一个变量定义

在vc中链接时就出现了i重复定义的错误, 而在c中成功编译。

结论:

(1). 当你第一个使用这个头的.cpp文件生成.obj的时候, int i 在里面定义了当另外一个使用这个的.cpp再次[单独]生成.obj的时候, int i 又被定义然后两个obj被另外一个.cpp也include 这个头的, 连接在一起, 就会出现重复定义。



把源程序文件扩展名改成.c后，VC按照C语言的语法对源程序进行编译，而不是C++。在C语言中，若是遇到多个int i，则自动认为第一个是定义，其他的是声明。

C语言和C++语言连接结果不同，可能（猜测）时在进行编译的时候，C++语言将全局量默认为强符号，所以连接出错。C语言则依照是否初始化进行强弱的判断的。（参考）

解决方法：

把源程序文件扩展名改成.c。

(2).推荐解决方案：

.h中只声明 extern int i;在.cpp中定义

```
<x.h>

#ifdef __X_H__

#define __X_H__

extern int i;

#endif // __X_H__

<x.c>

int i;
```

注意问题：

(1).变量一般不要定义在.h文件中。

ifndef/define/endif的用法与实例分析

用法：

.h文件，如下：

```
#ifndef XX_H
#define XX_H
.....
#endif
```

这样如果有两个地方都包含这个头文件，就不会出现两次包含的情况，因为在第二次包含时XX_H已经有定义了，所以就不再 include 了。

```
-----
-----

#ifndef GRAPHICS_H    // 防止graphics.h被重复引用
#define GRAPHICS_H

#include <math.h>      // 引用标准库的头文件
...

#include "myheader.h" // 引用非标准库的头文件
...

void Function1(...);   // 全局函数声明
...

class Box              // 类结构声明
{
...
};

#endif

-----
-----
```

假设你的工程里面有4个文件，分别是a.cpp, b.h, c.h, d.h

a.cpp的头部是：

```
#include "b.h"
#include "c.h"
```

b.h和c.h的头部都是：

```
#include "d.h"
```

而d.h里面有class D的定义。

这样一来，

编译器编译a.cpp的时候，先根据#include "b.h"去编译b.h这个问题，再根据b.h里面的#include "d.h"，去编译d.h的这个文件，这样就把d.h里面的class D编译了；然后再根据a.cpp的第二句#include "c.h"，去编译c.h，最终还是会找到的d.h里面的class D，但是class D之前已经编译过了，所以就会报重定义错误。

加上ifndef/define/endif，就可以防止这种重定义错误。

1. 比如你有两个C文件，这两个C文件都include了同一个头文件。而编译时，这两个C文件要一同编译成一个可运行文件，于是问题来了，大量的声明冲突。还是把头文件的内容都放在ifndef和endif中吧。

不管你的头文件会不会被多个文件引用，你都要加上这个。

一般格式是这样的：

```
#ifndef <标识>
#define <标识>
.....
.....
#endif <标识>
```

在理论上来说可以是自由命名的，但每个头文件的这个“标识”都应该是唯一的。标识的命名规则一般是头文件名全大写，前后加下划线，并把文件名中的“.”也变成下划线，如：stdio.h

```
#ifndef _STDIO_H_
#define _STDIO_H_
.....
#endif
```

2. 在ifndef中定义变量出现的问题（一般不定义在ifndef中）。

```
#ifndef AAA
#define AAA
...
int i;
...
#endif
```

里面有一个变量定义在vc中链接时就出现了i重复定义的错误，而在c中成功编译。

原因：

- (1). 当你第一个使用这个头的.cpp文件生成.obj的时候，int i 在里面定义了当另外一个使用这个的.cpp再次[单独]生成.obj的时候，int i 又被定义然后两个obj被另外一个.cpp也include 这个头的，连接在一起，就会出现重复定义。
- (2). 把源程序文件扩展名改成.c后，VC按照C语言的语法对源程序进行编译，而不是C++。在C语言中，若是遇到多个int i，则自动认为其中一个定义，其他的是声明。
- (3). C语言和C++语言连接结果不同，可能（猜测）时在进行编译的时候，C++语言将全局变量默认为强符号，所以连接出错。C语言则依照是否初始化进行强弱的判断的。

参考解决方法：

- (1). 把源程序文件扩展名改成.c。

(2). 推荐解决方案: .h中只声明 extern int i;

在. cpp中定义

```
#ifndef __X_H__
#define __X_H__

extern int i;

#endif // __X_H__ int i;
```

注意问题: 变量一般不要定义在.h文件中。

顶

0

踩

0

上一篇

NSNotificationCenter 的详细说明 --转

下一篇

UIWebView调用本地

我的同类文章

ios (42)

• Objective-C 对 URL 进行 U...

2013-07-17

阅读 6903

• UIScrollView 使用

2013-02-26

阅读 228

• 通过指定字符串文本大小判...

2013-02-21

阅读 493

• iOS 上的 VOIP 应用, 如何...

2012-12-28

阅读 408

• AVPlayer 多媒体播放器

2012-12-05

阅读 17066

• iphone ios 如何使用gcd, bl...

2012-11-09

阅读 558

• iOS ASIHTTPRequest详解

2013-03-26

阅读 346

• IOS socket使用大全

2013-02-25

阅读 457

• ios 后台长短时间运行

2013-01-07

阅读 2147

• iOS 后台程序误区 - 结束后...

2012-12-28

阅读 532

• 10个IOS开发第三方类库

2012-12-05

阅读 688

更多文章

猜你在找

- iOS程序员的C语言教程

VC++游戏开发基础系列从入门到精通

C语言及程序设计提高

C语言及程序设计初步

C语言及程序设计进阶
- Makefile详解

Linux下的makefile编写详解

Makefile详解自己觉得重新看一次学了好多东西红色字

LinuxUnix环境下的make和makefile详解

Makefile详解



免费的云主机



新加坡移民条件



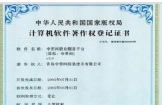
单身公寓



无限流量卡



档案管理系统



软件著作权

查看评论

暂无评论

您还没有登录,请[\[登录\]](#)或[\[注册\]](#)

* 以上用户言论只代表其个人观点，不代表CSDN网站的观点或立场

核心技术类目

全部主题 Hadoop AWS 移动游戏 Java Android iOS Swift 智能硬件 Docker OpenStack
VPN Spark ERP IE10 Eclipse CRM JavaScript 数据库 Ubuntu NFC WAP jQuery
BI HTML5 Spring Apache .NET API HTML SDK IIS Fedora XML LBS Unity
Splashtop UML components Windows Mobile Rails QEMU KDE Cassandra CloudStack FTC
coremail OPhone CouchBase 云计算 iOS6 Rackspace Web App SpringSide Maemo
Compuware 大数据 aptech Perl Tornado Ruby Hibernate ThinkPHP HBase Pure Solr
Angular Cloud Foundry Redis Scala Django Bootstrap

[公司简介](#) | [招贤纳士](#) | [广告服务](#) | [银行汇款帐号](#) | [联系方式](#) | [版权声明](#) | [法律顾问](#) | [问题报告](#) | [合作伙伴](#) | [论坛反馈](#)

[网站客服](#) [杂志客服](#) [微博客服](#) webmaster@csdn.net 400-600-2320 | 北京创新乐知信息技术有限公司 版权所有 | 江苏乐知网络技术有限公司 提供商务支持
京 ICP 证 09002463 号 | Copyright © 1999-2014, CSDN.NET, All Rights Reserved 