C++ 全局变量被自身文件/项目内其他文件/动态链接库(DLL)之外文件使用_全局变量不可在其他dll中可用c++-CSDN博客

成就一亿技术人!

全局变量的使用

一、全局变量的定义和基础使用

全局变量一般定义在一个.cpp文件的头部, 供所在文件乃至其它文件共享使用，供其它文件使用时，不能声明为静态static。

示例1：

```
Test1.cpp

#inclue <stdio.h>

char cTest1[32] = "Hello World!\n";
char cTest2[32] = "Hello My World!\n";

int print_test()
{
    printf("%s", cTest2);    //直接使用全局变量cTest2
    return 0;
}

int main()
{
  //打印出  Hello World!
    printf("%s", cTest1);            //直接使用全局变量cTest1

  //打印出 Hello My World!
    print_test();

    return 0;
}
```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23

二、全局变量被其它文件使用

全局变量被其它文件可访问，需要有两个基本条件：

①变量不能声明为静态，否则只能被全局变量所在文件访问，如下示例2是不可以的，按示例1处理即可。

示例2：

```
Test2.cpp

#inclue <stdio.h>


static char cTest1[32] = "Hello World!\n";            //由于使用static关键字修饰，因此只能被本.cpp的成员使用，其它文件不可见
static char cTest2[32] = "Hello My World!\n";


int print_test()
{
    printf("%s", cTest2);    //直接使用全局变量cTest2
    return 0;
}


int main()
{
  //打印出  Hello World!
    printf("%s", cTest1);            //直接使用全局变量cTest1


  //打印出 Hello My World!
    print_test();

    return 0;
}
```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27

②定义全局变量对应的头文件中或其它使用全局变量的文件中，使用extern关键字声明全局变量

(1) 在.h文件中声明

示例3（全局变量的定义如示例1）：

```
Test1.h

extern char char cTest1[32];
extern char char cTest2[32];
```

1
2

```
3
4
```

```
Test3.cpp

#include <stdio.h>
#include "Test1.h"    //inculde 被extern 修饰的全局变量所在的头文件

int print_test3()
{
    printf("%s",cTest1);   // 使用全局变量 //当程序被执行时 打印出  Hello World!
    printf("%s",cTest2);   // 使用全局变量 //当程序被执行时 打印出 Hello My World!
    return 0;
}
```

```
1
2
3
4
5
6
7
8
9
10
11
```

(2) 在调用全局变量的文件中声明

示例4（全局变量的定义如示例1）：

```
Test:4.cpp

#include <stdio.h>

extern char char cTest1[32];
extern char char cTest2[32];

int print_test3()
{
    printf("%s",cTest1);   // 使用全局变量 //当程序被执行时 打印出  Hello World!
    printf("%s",cTest2);   // 使用全局变量 //当程序被执行时 打印出 Hello My World!
    return 0;
}
```

```
1
2
3
4
5
6
7
8
9
10
11
12
13
```

本人的使用习惯是二者同时使用，既在头文件中声明，也在调用的文件中声明。

三、全局变量在动态链接库中定义，被外部文件使用

动态链接库中的全局变量被其它文件可访问，需要有三个基本条件：
①②同上一节中
③在全局变量定义文件中，使用__declspec(dllexport) 声明输出；在全局变量调用的外部文件中，使用__declspec(dllimport)声明输入

针对条件③进行说明：

```
方便起见，使用如下方式，通过定义宏的方式，定义上面的导入和导出声明

#ifdef _LIBDATAPRO1_EXPORTS    //该宏在VS中， “项目”→“右键”→“属性”→“属性配置”→“C/C++”→“预处理器”→“预处理器定义” 中添加
#define LIBCFGPARA_API __declspec(dllexport)    //若 _LIBDATAPRO1_EXPORTS  已定义 则  LIBCFGPARA_API 定义为输出  __declspec(dllexport)
#else
#define LIBCFGPARA_API __declspec(dllimport)    //若 _LIBDATAPRO1_EXPORTS  未定义 则  LIBCFGPARA_API 定义为输   __declspec(dllimport)
#endif // _LIBDATAPRO_EXPORT
```

示例5：

```
Test5.cpp

#inclue <stdio.h>

LIBCFGPARA_API char cTest1[32] = "Hello World!\n";            //使用 LIBCFGPARA_API 声明全局变量为输出形式  LIBCFGPARA_API 为  __declspec(dllexport)
LIBCFGPARA_API char cTest2[32] = "Hello My World!\n";         //使用 LIBCFGPARA_API 声明全局变量为输出形式  LIBCFGPARA_API 为  __declspec(dllexport)
```

```
Test5.h

#ifdef _LIBDATAPRO1_EXPORTS    //该宏在VS中，“项目”→“右键”→“属性”→“属性配置”→“C/C++”→“预处理器”→“预处理器定义”中添加
#define LIBCFGPARA_API __declspec(dllexport)    //若 _LIBDATAPRO1_EXPORTS  已定义 则  LIBCFGPARA_API 定义为输出  __declspec(dllexport)
#else
#define LIBCFGPARA_API __declspec(dllimport)    //若 _LIBDATAPRO1_EXPORTS  未定义 则  LIBCFGPARA_API 定义为输   __declspec(dllimport)
#endif // _LIBDATAPRO_EXPORT


LIBMAKEPRE_API extern char char cTest1[32];               //使用 LIBCFGPARA_API 配合 extern 声明全局变量为输出形式  LIBCFGPARA_API 为  __declspec(dllexport)
LIBMAKEPRE_API extern char char cTest2[32];               //使用 LIBCFGPARA_API 配合 extern 声明全局变量为输出形式  LIBCFGPARA_API 为  __declspec(dllexport)
```

将上述Test5项目(含Test5.cpp 和 Test5.h文件)进行编译，生成动态链接库，得到Test5.dll和Test5.lib两文件。

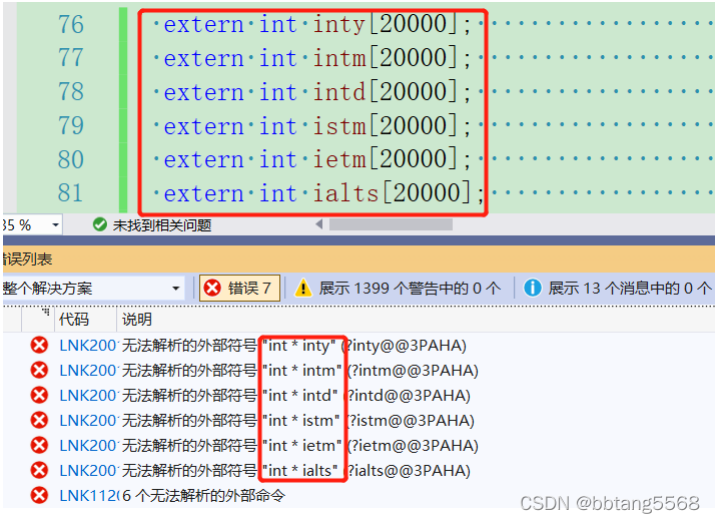
```
Test6.cpp

#include <stdio.h>
#inclue "Test5.h"                                        //(1) include 全局变量对应的Test5.h文件。      由于外部文件的项目中，并没有定义 _LIBDATAPRO1_EXPORTS，因此
LIBCFGPARA_API 被 #else 分支 定义为 __declspec(dllimport)

#pragma comment(lib, "../x64/Debug/Test5.lib")           //(2) 引用编译生成的.lib库文件

LIBMAKEPRE_API extern char char cTest1[32];              //(3) 使用 LIBCFGPARA_API 配合 extern 声明全局变量为输出形式  LIBCFGPARA_API 为  __declspec(dllimport)
LIBMAKEPRE_API extern char char cTest2[32];              //(3) 使用 LIBCFGPARA_API 配合 extern 声明全局变量为输出形式  LIBCFGPARA_API 为  __declspec(dllimport)

int print_test()
{
    printf("%s", cTest2);    //直接使用全局变量cTest2
    return 0;
}

int main()
{
  //打印出  Hello World!
    printf("%s", cTest1);            //直接使用全局变量cTest1


  //打印出 Hello My World!
    print_test();
```

```
    return 0;
}
```

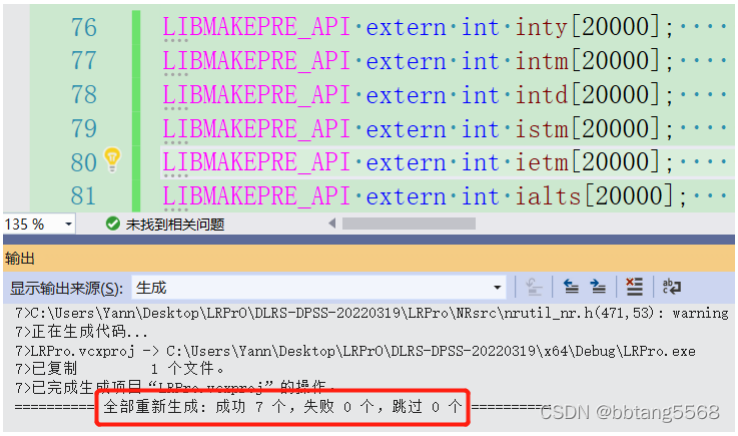
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27

注意：若外部文件在在使用全局变量时， 没有使用输入声明 __declspec(dllimport)，则VS编译时会有“无法解析的外部符号”编译错误，对比结果如下图所示。

示例6（无 __declspec(dllimport) 声明，编译出错）：



示例6（有 __declspec(dllimport) 声明，编译成功）：



至此，动态链接库中的全局变量，实现了被外部文件的使用。