<https://blog.csdn.net/zhuce0001/article/details/20651025>

本文主要讲解在QT开发环境中如何使用VC生成的DLL及QT自身生成的DLL。至于其它情况本文不作讨论。

连接库分为2种

（1）动态连接库，通常有.h .lib .dll三个文件，功能实现在dll中

（2）静态连接库，通常有.h .lib二个文件，功能实现在lib中

由上可以看出动态库的lib和静态库的lib文件是不同的。

    如果使用生成连接库的开发环境与使用连接库的开发环境相同，一般不会出什么问题，如VC写的连接库

（包括动态库和静态库）还在VC中用一般不会有什么问题的。有时候我们需要DLL跨开发环境，如以前VC下的

DLL想在QT中用。有网友说QT不支持VC生成的静态库，所以只测试QT使用动态库的情况。

【先看VC生成的DLL】

使用VC6建立Win32 dynamic-link library工程，工程中只有两个文件，编译即可生成DLL和LIB

**[html]** [view plain](https://blog.csdn.net/zhuce0001/article/details/20651025) [copy](https://blog.csdn.net/zhuce0001/article/details/20651025)

1. ////////////////////////////////// MFCDLL.h  /////////////////////////////////////
2. #ifndef \_MFCDLL\_H
3. #define \_MFCDLL\_H
5. #ifdef \_\_cplusplus
6. extern "C" {
7. #endif
9. #ifdef DLL
10. // do nothing
11. #else
12. #define DLL \_\_declspec(dllimport)
13. #endif
15. DLL void hello();
16. DLL int add(int a, int b);
18. #ifdef \_\_cplusplus
19. }
20. #endif
22. #endif

**[cpp]** [view plain](https://blog.csdn.net/zhuce0001/article/details/20651025) [copy](https://blog.csdn.net/zhuce0001/article/details/20651025)

1. ////////////////////////////////// MFCDLL.cpp  /////////////////////////////////////
2. #define DLL \_\_declspec(dllexport)
3. #include "MFCDLL.h"
4. #include <windows.h>
6. **void** hello()
7. {
8. ::MessageBox(NULL, "hello world!",
9. "greeting", MB\_OK);
10. }
12. **int** add(**int** a, **int** b)
13. {
14. **return** a + b;
15. }

【使用QT生成DLL】使用QT建立动态库工程，编译即可得到DLL（无LIB文件）

**[cpp]** [view plain](https://blog.csdn.net/zhuce0001/article/details/20651025) [copy](https://blog.csdn.net/zhuce0001/article/details/20651025)

1. ////////////////////////////////// qtdll\_global.h  //////////////////////////////
2. #ifndef QTDLL\_GLOBAL\_H
3. #define QTDLL\_GLOBAL\_H
5. #include
7. #if defined(QTDLL\_LIBRARY)
8. #  define QTDLLSHARED\_EXPORT Q\_DECL\_EXPORT
9. #else
10. #  define QTDLLSHARED\_EXPORT Q\_DECL\_IMPORT
11. #endif
13. #endif // QTDLL\_GLOBAL\_H

**[cpp]** [view plain](https://blog.csdn.net/zhuce0001/article/details/20651025) [copy](https://blog.csdn.net/zhuce0001/article/details/20651025)

1. ////////////////////////////////// qtdll.h  /////////////////////////////////////
2. #ifndef QTDLL\_H
3. #define QTDLL\_H
5. #include "qtdll\_global.h"
7. **class** QTDLLSHARED\_EXPORT QTDLL
8. {
9. **public**:
10. QTDLL();
12. **public**:
13. **int** add(**int** a, **int** b);
14. };
16. #endif // QTDLL\_H

**[cpp]** [view plain](https://blog.csdn.net/zhuce0001/article/details/20651025) [copy](https://blog.csdn.net/zhuce0001/article/details/20651025)

1. ////////////////////////////////// qtdll.cpp  ///////////////////////////////////
2. #include "qtdll.h"

5. QTDLL::QTDLL()
6. {
7. }
9. **int** QTDLL::add(**int** a, **int** b)
10. {
11. **return** a + b;
12. }

【QT显式加载VC生成的DLL】

**[cpp]** [view plain](https://blog.csdn.net/zhuce0001/article/details/20651025) [copy](https://blog.csdn.net/zhuce0001/article/details/20651025)

1. #include "mainwindow.h"
2. #include "ui\_mainwindow.h"
3. #include <QLibrary>
4. #include <QDebug>
6. MainWindow::MainWindow(QWidget \*parent) :
7. QMainWindow(parent),
8. ui(**new** Ui::MainWindow)
9. {
10. ui->setupUi(**this**);
12. // DLL显式加载，只需要DLL文件即可，不需要.H和.LIB文件
13. // 需要将DLL放到可执行目录中
14. **typedef** **void**(\*FUN1)();
15. **typedef** **int**(\*FUN2)(**int**, **int**);
17. QLibrary lib("MFCDLL.dll");
18. **if** (lib.load()) {
19. qDebug() << "load ok!";
21. FUN1 hello = (FUN1)lib.resolve("hello");
22. FUN2 add = (FUN2)lib.resolve("add");
23. **if** (hello) {
24. qDebug() << "load hello ok!";
25. hello();
26. }
27. **if** (add) {
28. qDebug() << "load add ok!";
29. qDebug() << add(3, 5);
30. }
31. } **else** {
32. qDebug() << "load error!";
33. }
34. }

【QT隐式加载VC生成的DLL】

**[cpp]** [view plain](https://blog.csdn.net/zhuce0001/article/details/20651025) [copy](https://blog.csdn.net/zhuce0001/article/details/20651025)

1. #include "mainwindow.h"
2. #include "ui\_mainwindow.h"
3. #include "lib/MFCDLL.h"
4. #include <QDebug>
6. MainWindow::MainWindow(QWidget \*parent) :
7. QMainWindow(parent),
8. ui(**new** Ui::MainWindow)
9. {
10. ui->setupUi(**this**);
12. // DLL隐式加载，只需要.DLL .H和.LIB文件
13. // 1需要将DLL放到可执行目录中
14. // 2将LIB路径设置到项目PRO文件中
15. // 3将头文件包含进来，如果不包含需要自已声明函数原型及来源（本质与包含头文件相同）
16. hello();
17. qDebug() << add(5, 6);
18. qDebug() << "ok";
19. }

pro工程文件中要设置LIB文件路径

**[cpp]** [view plain](https://blog.csdn.net/zhuce0001/article/details/20651025) [copy](https://blog.csdn.net/zhuce0001/article/details/20651025)

1. # lib文件路径
2. LIBS += "F:/lib/MFC\_DLL\_TEST\_WITH\_QT\_2/lib/MFCDLL.lib"

【QT使用QT生成的动态库，隐式】

**[cpp]** [view plain](https://blog.csdn.net/zhuce0001/article/details/20651025) [copy](https://blog.csdn.net/zhuce0001/article/details/20651025)

1. #include "mainwindow.h"
2. #include "ui\_mainwindow.h"
3. #include <QDebug>
4. #include "lib/qtdll.h"
6. MainWindow::MainWindow(QWidget \*parent) :
7. QMainWindow(parent),
8. ui(**new** Ui::MainWindow)
9. {
10. ui->setupUi(**this**);
12. // QT使用QT生成的DLL
13. // 1. 包含头文件
14. // 2. 在工程文件中指定lib路径
15. // 3. 将动态库拷贝到可执行文件目录
16. QTDLL dll;
17. qDebug() << dll.add(3, 5);
18. }

pro工程文件中的设置

**[cpp]** [view plain](https://blog.csdn.net/zhuce0001/article/details/20651025) [copy](https://blog.csdn.net/zhuce0001/article/details/20651025)

1. LIBS += "F:/lib/QT\_DLL\_TEST\_WITH\_DLL/lib/QTDLL.dll"