# 1.生成dll，创建一个项目，选择library

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

然后点击next-》next-》ok，会生成如下的文件

|  |
| --- |
|  |

#### 右击Sources文件夹，-》新建-》选择Qt-》qt resourcefile（qrc）

|  |
| --- |
|  |

然后填写名称

|  |
| --- |
|  |

ok-》ok出现如下界面

|  |
| --- |
|  |

#### 关闭这个界面，并且右击生成的Plugin.qrc，添加Resource文件，选择Qml File

|  |
| --- |
|  |

填写qml文件的名称-》ok

|  |
| --- |
|  |

点击下一步，再点击完成，面如下：

|  |
| --- |
|  |

#### 在MyButton.qml中添加代码

|  |
| --- |
| import QtQuick 2.0  import QtQuick.Controls 2.15  Button{  width: 100  height: 60  text: "Button1"  onClicked: *console*.log("hello,you clicked me...")  } |

#### 在myqmlwidget\_plugin.cpp中注册控件给外部使用

|  |
| --- |
| #include "myqmlwidget\_plugin.h"  #include "myqmlwidget.h"  #include <qqml.h>  void MyQmlwidgetPlugin::***registerTypes***(const char \***uri**)  {  // @uri MyQmlWidget  qmlRegisterType<MyQmlWidget>(uri, 1, 0, "MyQmlWidget");  qmlRegisterType(QUrl("qrc:/MyButton.qml"),uri,1,0,"MyButton");  } |

#### 然后生成dll，点击QtCreator的项目按钮，取消勾选shadow build选项，然后点击build生成dll

# 2、使用生成的dll

创建一个qt quick2项目，叫做testQmlDll

|  |
| --- |
|  |

#### 先编译一下这个项目，取消勾选shadow build选项，然后点击build生成，这样子输出文件夹也在源码文件夹里面，如下图

|  |
| --- |
|  |

#### 在这个目录的dubug项目项目创建一个文件夹，名字必须和控件的名称一样

|  |
| --- |
|  |

#### 将上面生成的dll和qmldir(在输出目录的根目录下面)文件拷贝到这个目录下面

|  |
| --- |
|  |

#### 在test项目在使用

|  |  |
| --- | --- |
| import QtQuick 2.15  import QtQuick.Window 2.15  import MyQmlWidget 1.0  Window {  width: 640  height: 480  visible: true  title: *qsTr*("Hello World")  MyButton{  }  } |  |

注意将Qml控件封装成dll后，在使用它的地方就找不到他的属性，这个和直接在qml中使用的控件还是不一样的，必须将所有需要的功能和属性都封装在里面，否则找不到！！！

其实将控件写成qml文件然后直接使用更好！！！，当然，**如果你的控件的功能很强大而你又不想其他人知道怎么实现的，就可以将它封装为dll。**