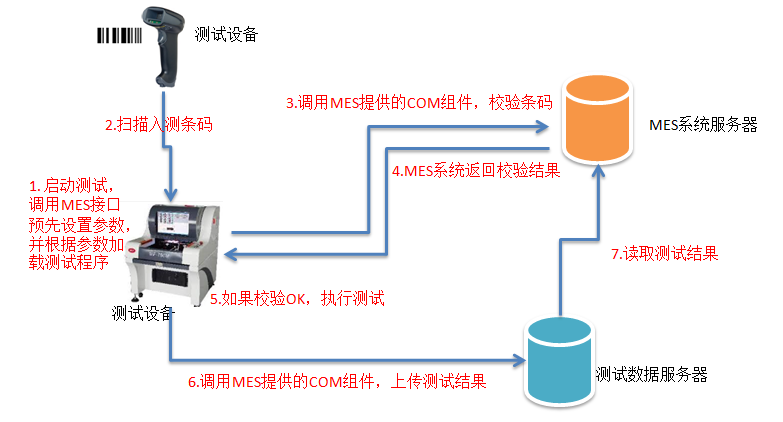
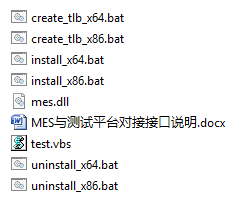
**MES系统与测试平台对接说明**

# 1. 对接原理



# 2. MES提供的COM组件

MES提供的COM组件，包含以下文件：



mes.dll，COM组件本身

install\_x86.bat和install\_x64.bat，用于安装COM组件。

uninstall\_x86.bat和uninstall\_x64.bat，用于卸载COM组件。

create\_tlb\_x86.bat和create\_tlb\_x64.bat，用于创建注册COM组件的TLB文件。

test.vbs，用于测试COM组件是否安装成功。

# 3.使用步骤

1. 先确保已经安装Microsoft .NET Framework 4.0，本COM组件基于.NET 4.0创建。

下载地址：

<https://download.microsoft.com/download/9/5/A/95A9616B-7A37-4AF6-BC36-D6EA96C8DAAE/dotNetFx40_Full_x86_x64.exe>

二. 执行install\_x86.bat或install\_x64.bat安装。

三. 执行test.vbs，如果能正确执行，则表示COM组件安装成功。

四. 测试平台开发商导入mes.tlb文件，按照下一节描述的接口调用即可。

五. 执行uninstall.bat，卸载COM组件

**注意：如果测试平台是基于.NET Framework 4.0开发的，不需要以COM组件方式调用，可以直接引用mes.dll程序集， 调用下一节描述的类型和方法即可。**

# 4. 函数接口说明

mes.dll组件提供IMesService接口和MesService类，其定义如下（C#语法）：

## IMesService接口:

[Guid("EBB7879B-662B-4938-9908-C708A5278D38")]

public interface IMesService {

string GetOrder();

string GetProgram(string type);

bool CheckBarcode(string barcode);

void SaveTestData(string barcode, string result, string details, string file);

}

## MesService类:

[Guid("E8BE778C-734D-4786-9AB5-A8028DE74F42")]

public class MesService: IMesService {

public string GetOrder() { }

public string GetProgram(string type) { }

public bool CheckBarcode(string barcode) { }

public void SaveTestData(string barcode,string result,string details,string file)

}

## 四个函数的说明：

**注意：由于运行期间四个函数有共享数据，要求整个测试平台运行期间，只能创建一个MesService组件对象实例。也就是不能每次调用一个函数就创建一个组件对象。**

|  |  |
| --- | --- |
| string GetOrder() | |
| **输入参数** | 无。 |
| **返回值** | 返回工单号，字符串。如果返回NULL或空字符串，则不能进行测试。 |
| **调用说明** | 调用此函数，会打开大致如下的窗口：    点确定，返回选择的工单编号。  点取消，函数返回NULL |

|  |  |
| --- | --- |
| string GetProgram(string type) | |
| **输入参数** | type，脚本类型，用于区分不同测试平台。建议传入测试脚本的扩展名，例如dat，prg等。 |
| **返回值** | 如果返回NULL或空字符串，则不能进行测试。  如果返回"USE\_DEFAULT\_PROGRAM"，则使用测试平台本身打开的测试程序。  否则，按照返回的程序名加载测试程序。 |
| **调用说明** | 测试平台打开或加载测试脚本时，调用此函数，获取测试脚本的全路径文件名。  调用此函数，会弹出如下界面（实际可能不同，但不影响接口调用）：    点确定，返回”USE\_DEFAULT\_PROGRAM”或全路径文件名，两种情况调用方都要正确处理。  点取消，函数返回NULL |

|  |  |
| --- | --- |
| bool CheckBarcode(string barcode) | |
| **输入参数** | barcode，入测条码 |
| **返回值** | bool，是否可以入测 |
| **调用说明** | 用户扫描条码后，测试平台调用此函数，检查条码是否可以入测。  如果返回true，则可以进行测试。如果返回false则不进行测试。函数内部会进行错误提示。 |

|  |  |
| --- | --- |
| void SaveTestData(string barcode, string result, string details, string file) | |
| **输入参数** | barcode，入测条码  result，测试结果，PASS或FAIL  details，XML字符串。  file，测试结果存储文件路径，全路径文件名，例如d:\test\_data\123.txt |
| **返回值** | 无。 |
| **调用说明** | 测试完成后，不论测试结果PASS或FAIL，都调用此函数上传测试结果。 |

关于SaveTestData 函数的details参数，不同的设备类型，请按照下表格式描述提供：

|  |  |
| --- | --- |
| 设备类型 | 数据格式 |
| AOI设备 | <faults>  <item line="1" pos="位置号，例如R11" cp="器件编号，例如E10201001" md="器件型号" ft="缺陷描述，例如虚焊" rmk="描述" rslt="结果(P/F)"/>  <item line="2" pos="R12" cp="E10201001" md="" ft="虚焊" rmk="" rslt="F"/>  </faults> |
| ATE设备 | <inspections sw\_ver1=”” sw\_ver2=”” sw\_ver3=”” sw\_ver4=”” sw\_ver5=””>  <item line="1"  name="测试项目1" rmk="描述" cp="器件编号" std="标准值" max="上限" min="下限" act="实测值" um="单位" cd="条件" rslt="判定结果(P/F)"/>  <item line="2"  name="测试项目1" rmk="描述" cp="器件编号" std="标准值" max="上限" min="下限" act="实测值" um="单位" cd="条件" rslt="判定结果(P/F)"/>  <item line="3"  name="测试项目1" rmk="描述" cp="器件编号" std="标准值" max="上限" min="下限" act="实测值" um="单位" cd="条件" rslt="判定结果(P/F)"/>  </ inspections> |

该XML字符串实际拼接时可以不加换行，上述示例为了直观显示，所以加了换行。

拼接XML字符串时，请先将字段值中的特殊字符处理掉，否则可能因为值包含特殊字符，导致生成的XML格式无效。参考以下C#代码，改写成相应语法调用即可：

public string EscapeAttributeValue(string value) {

    return value.Replace("&", "&amp;").Replace("<", "&lt;").Replace(">", "&gt;").Replace("\"", "&quot;").Replace("\r", "&#xD;").Replace("\n", "&#xA;");

}

也就是按下表顺序，将左边字符替换为右边的字符串:

|  |  |
| --- | --- |
| & | &amp; |
| < | &lt; |
| > | &gt; |
| " | &quot; |
| \r | &#xD; |
| \n | &#xA; |

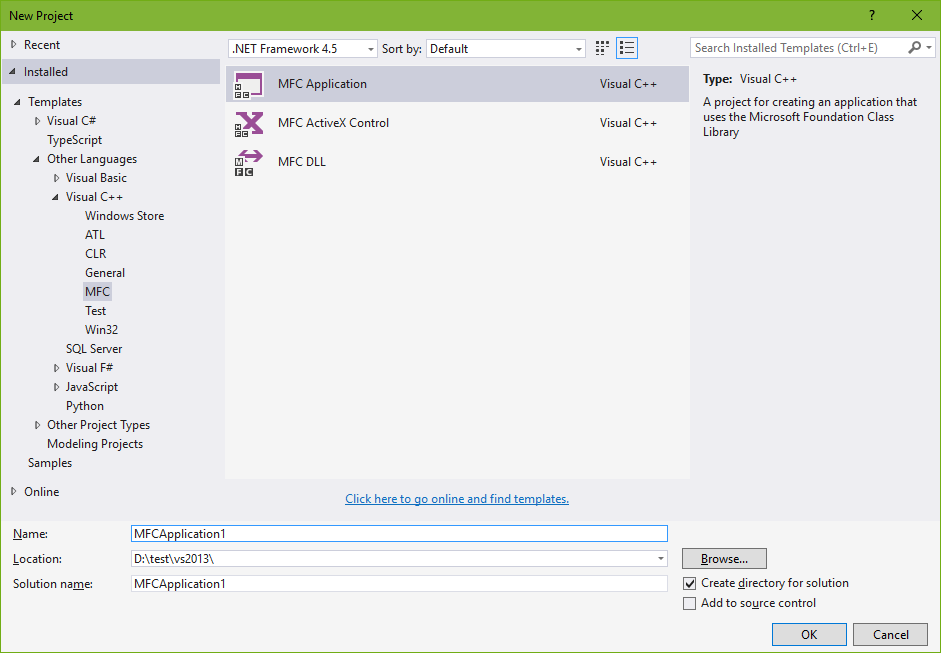
例如，拼接测试项目的时候，假定：

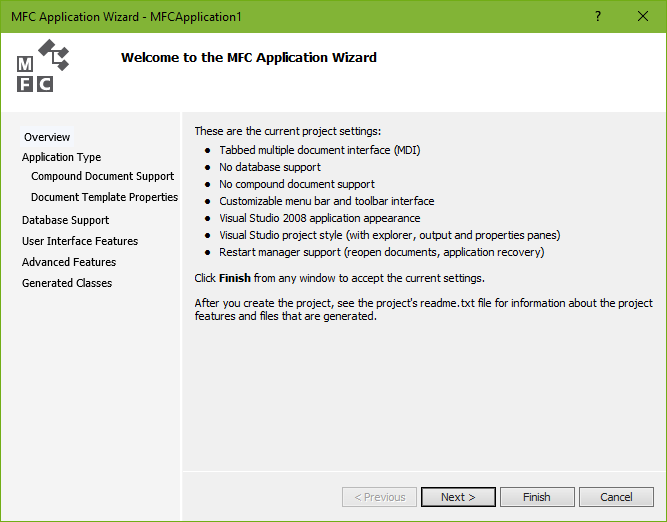
name="&测试项目1—电阻必须<100"

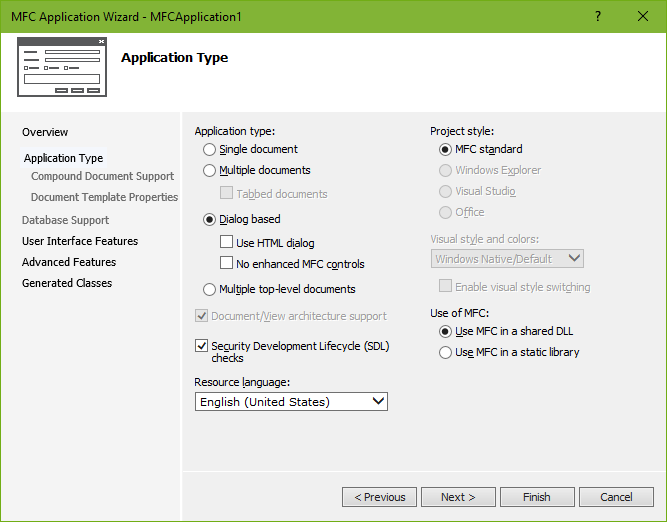
因为name中包含了XML的特殊字符“&”和“<”，所必须先处理一下：

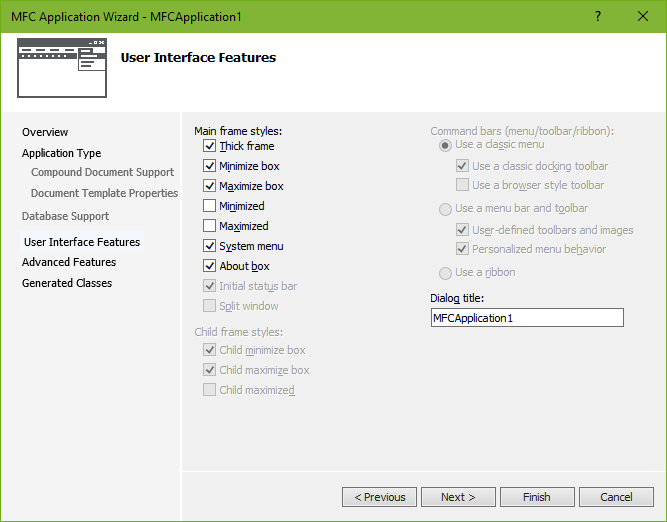
name=EscapeAttributeValue(name)

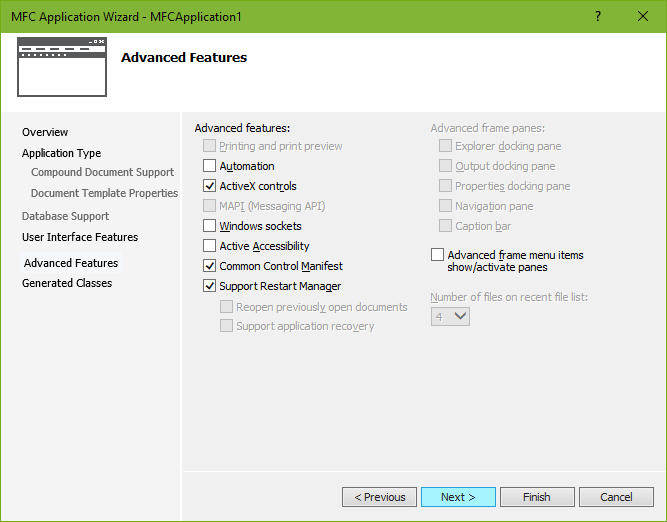
# 5. MFC调用示例

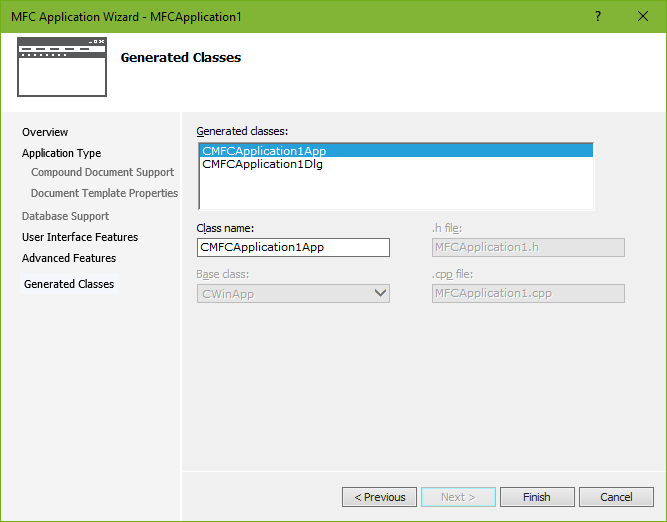


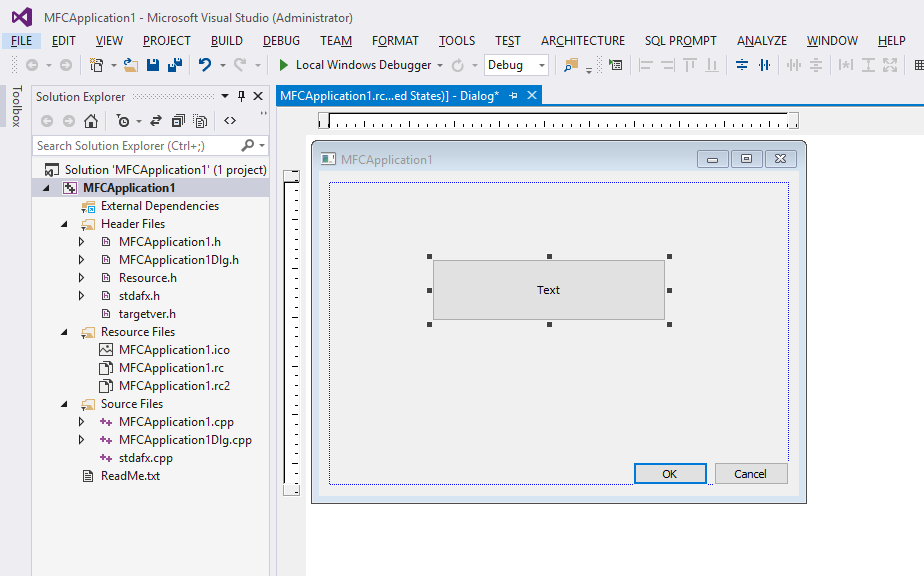


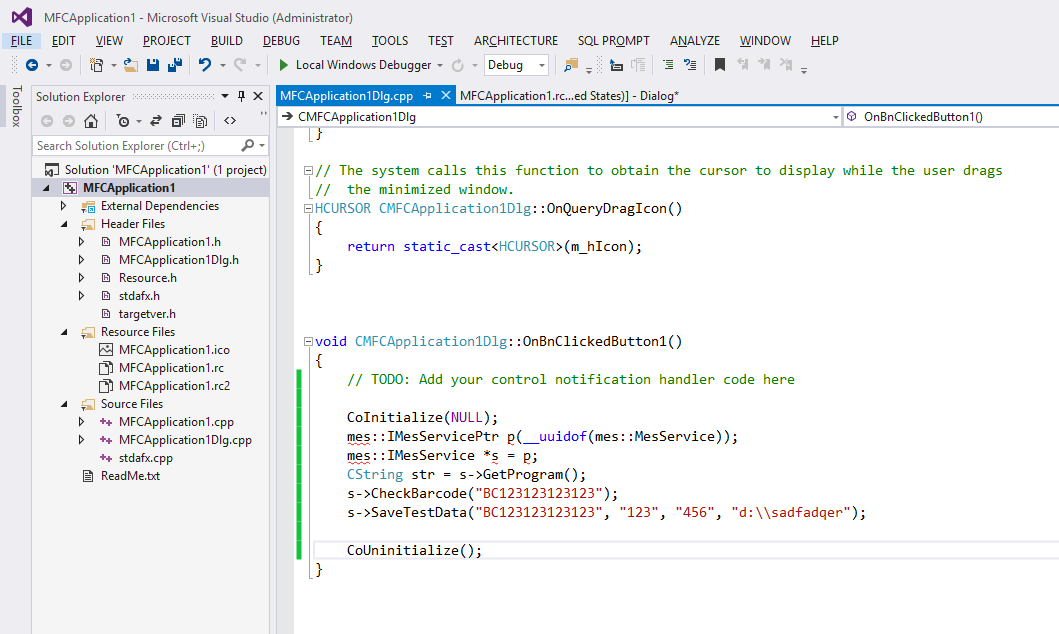


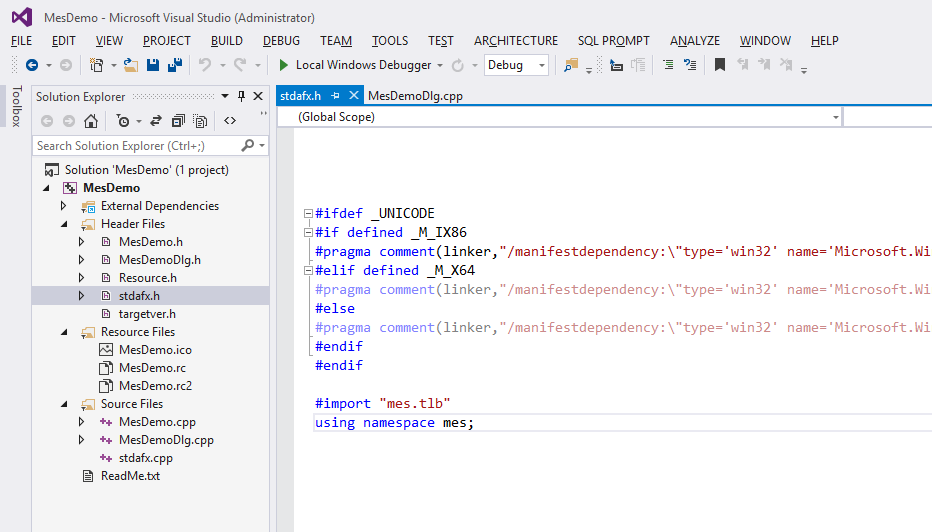


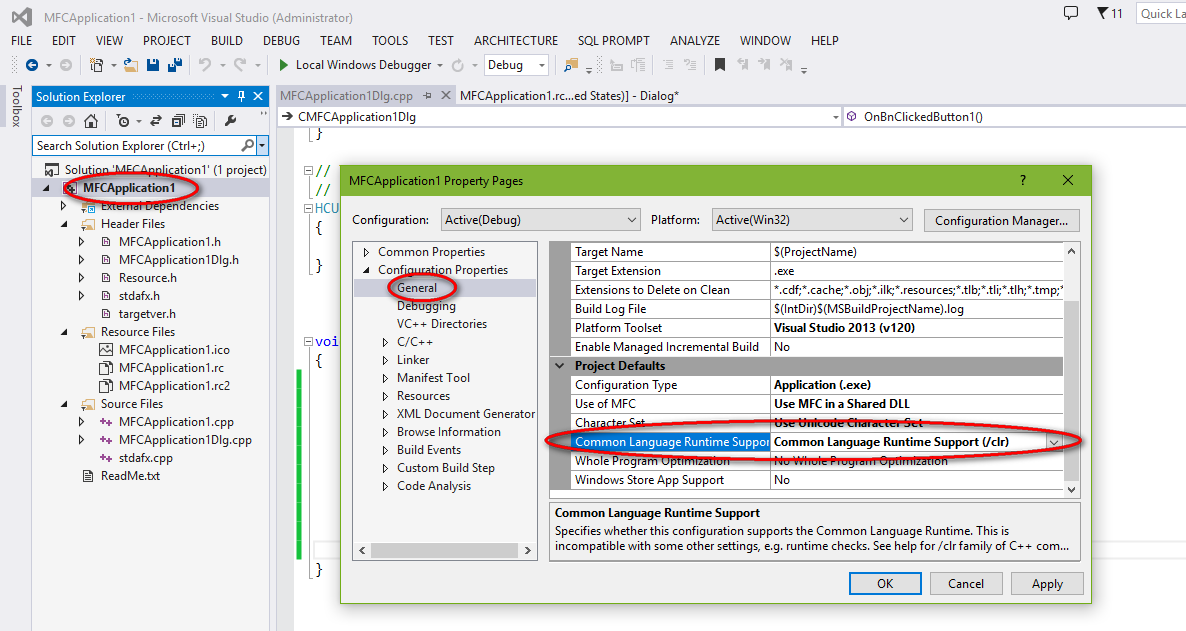


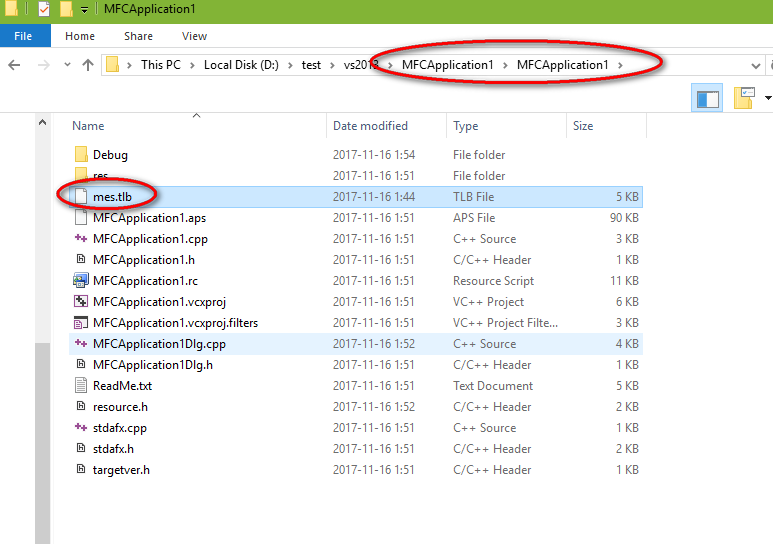












然后项目右键点Build，调试运行即可。