COM实用入门教程　　第一讲

主讲人：阙海忠

VC知识库网站 (www.vckbase.com) 拍摄制作

本讲要点：

- 一、现实中的组件与接口；
- 二、把现实中的思想融入到软件中；
- 三、C++程序中的组件与接口；
- 四、COM组件与COM接口；
- 五、QueryInterface函数，HRESULT类型，IID类型，数据类型转换。

现实中的组件与接口

组件的定义：含有可独立性，可通用性，可组合性，可替换性的事物，我们把它称作组件。

现实世界中存在各种各样的具备组件概念的设备。如：电脑主机内的设备： CPU、内存条、硬盘、光驱。还比如：5号电池、7号电池、插座。

为什么说这些设备具备组件概念呢？因为这些设备都具有可独立性，可通用性，可组合性，可替换性。

现实中的组件与接口

接口的定义：是组件与组件之间，组件与外部事物之间进行交互的协议。

组件与组件，组件与其它设备的交互工作是通过接口进行的。 CPU跟主板之间有固定的接口，内存条跟主板之间也有固定的接口。如果CPU要更换，被更换的CPU必须与旧的CPU拥有相同的使用接口。

现实中的组件与接口

* 主板不直接认识CPU，只认识CPU的接口。主板也不直接认识内存条，只认识内存条的接口。
* 所以接口的约定是很重要的。在组件的开发之前，必须先约定组件与外界交互的接口协议。
* 接口协议只要确定后，往往是不能再改变的，比如5号电池不能再做得大一点或再小一点，然后去替换旧的5号电池。

现实中的组件与接口

采用组件与接口的思想来开发设备，体现了社会分工的一个现象，也是社会生产力发展的必然过程，做CPU的厂商只管做CPU，不用告诉主板关于CPU本身的实现细节，也不用去了解主板的实现细节。

这一讲，主要讲解如下要点：

- 一、现实中的组件与接口；
- 二、把现实中的思想融入到软件中；
- 三、C++程序中的组件与接口；
- 四、COM组件与COM接口；
- 五、QueryInterface函数，HRESULT类型，IID类型，数据类型转换。

把现实中的思想融入到软件中

面向对象思想来源于大自然，让我们面向着是一个一个的对象，不是面向一个一个的过程（面向过程思想）。

黑格尔说“存在的就是合理的”，大自然存在各种各样的对象，每类对象具有自己的特性，对象存在继承关系。这些存在是合理的。最后把这些合理的思想演变成面向对象思想，所以面向对象思想是合理的。

把现实中的思想融入到软件中

社会的发展促使人类在劳动上的分工，分工又以约定的接口协议来交互。社会的发展产生了这种“组件-接口”的开发思想，这种思想又是长期没有被替换过的思想，这是一种好的思想，合理的思想。

若我们以组件方式架构我们的软件。我们软件中的组件将具有可独立性，可通用性，可组合性，可替换性；我们的软件也将具有更好的灵活性，可拓展性和可维护性。我们软件的开发过程也会变得更加的简单，更好的分工，更加的规范。

这一讲，主要讲解如下要点：

- 一、现实中的组件与接口；
- 二、把现实中的思想融入到软件中；
- 三、C++程序中的组件与接口；
- 四、COM组件与COM接口；
- 五、QueryInterface函数，HRESULT类型，IID类型，数据类型转换。

C++程序中的组件与接口

- 接口，是一种约定，一种协议。它是抽象的，指明了具体含义，但却没有实现这个定义。
  我们看一下C++的纯虚函数：求最大公约数，virtual int GreatestCommonDivisor(int a, int b) = 0; //求a与b的最大公约数。

这个函数的定义很明确，但没有实现这个含义的具体方法，所以，是抽象的。

C++程序中的组件与接口

- 我们一般采用interface这个英文单词表示C++中的接口，它在Microsoft Visual Studio 安装目录\VC\PlatformSDK\include\objbase.h中被预定义。

```
#define   interface   struct
```

- 在其它开发平台下，也可以自己编写预定义代码。

C++程序中的组件与接口

```
//接口IX
interface IX
{
    virtual void Fx1() = 0;
    virtual void Fx2() = 0;
};

//接口IY
interface IY
{
    virtual void Fy1() = 0;
    virtual void Fy2() = 0;
};
```

C++程序中的组件与接口

组件是派生于接口的，现在我们定义一个组件CA，它派生于IX和IY

```
//组件CA
class CA: public IX, public IY
{
public:
    virtual void Fx1() { cout << "Fx1" << endl; }

    virtual void Fx2(){ cout << "Fx2" << endl; }

    virtual void Fy1() {cout << "Fy1" << endl; }

    virtual void Fy2() { cout << "Fy2" << endl; }
};
```

C++程序中的组件与接口

客户端的调用查看Section1Demo1

这一讲，主要讲解如下要点：

- 一、现实中的组件与接口；
- 二、把现实中的思想融入到软件中；
- 三、c++程序中的组件与接口；
- 四、COM组件与COM接口；
- 五、QueryInterface函数，HRESULT类型，IID类型，数据类型转换。

COM组件与COM接口

- COM的定义：是Component Object Model （组件对象模型）的缩写
- COM组件是可以以二进制的形式发布，具有指定规则的二进制结构；
- COM组件是可以被其它应用程序来调用，以实现二进制代码的共享（跨应用）；
- COM组件是完全与编程语言无关的。(跨语言)；
- COM组件只能被运行在Windows操作系统平台上面，Linux，Mac不能适用。

COM组件与COM接口

- COM组件的内存结构和C++编译器为抽象基类所生成的内存结构是相同的。因此可以用C++的抽象基类来定义COM接口。
- COM组件必须继承于最基本的COM接口：IUnknow。
- IUnknow有三个函数，为别是QueryInterface, AddRef, Release。

COM组件与COM接口

```
interface IUnknown
{
        virtual HRESULT QueryInterface(const IID
  &iid, void **ppv) = 0;
        virtual ULONG AddRef() = 0;
        virtual ULONG Release() = 0;
};
```

这一讲，主要讲解如下要点：

- 一、现实中的组件与接口；
- 二、把现实中的思想融入到软件中；
- 三、c++程序中的组件与接口；
- 四、COM组件与COM接口；
- 五、QueryInterface函数，HRESULT类型，IID类型，数据类型转换。

QueryInterface

可以通过QueryInterface函数来查询某个组件是否支持某个特定的接口。若支持，QueryInterface返回一个指向此接口的指针。

这里我们看到函数返回类型为HRESULT，参数其中一个的类型是const IID&。

HRESULT跟IID是什么呢？

HRESULT

	A	B
1	HRESULT	含义
2	S_OK	成功
3	S_FALSE	函数成功执行完成，但返回时出现错误
4	E_INVALIDARG	参数有错误
5	E_OUTOFMEMORY	内存申请错误
6	E_UNEXPECTED	未知的异常
7	E_NOTIMPL	未实现功能
8	E_FAIL	没有详细说明的错误。
9	E_POINTER	无效的指针
10	E_HANDLE	无效的句柄
11	E_ABOUT	终止操作
12	E_ACCESSDENIED	访问被拒绝
13	E_NOINTERFACE	不支持接口

IID

- IID，接口标识符，每个接口都可以设置一个IID，用于标志该接口，若标志了某个接口后，IID的值不能再修改。
- IID其实是：

typedef GUID IID;

GUID

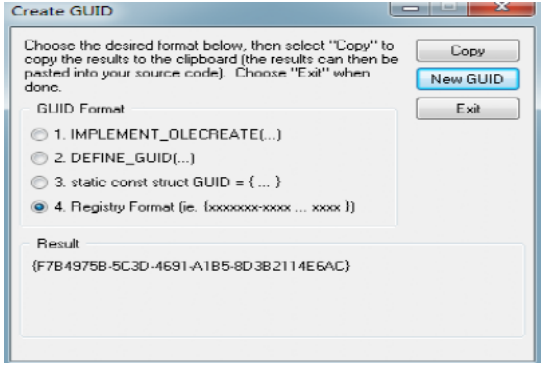
```
typedef struct _GUID
{
        DWORD  Data1;    //随机数
        WORD  Data2;              //和时间相关
        WORD  Data3;              //和时间相关
        BYTE  Data4[8];    //和网卡MAC相关
} GUID;
```

GUID

- GUID有16个字节，共128位二进制数。
- GUID的生成方法，可以采用Windows SDK v6.0A的Tools文件夹下的GUID生成器生成。

- 从理论上讲，它是不能保证唯一，但由于重复的可能性非常非常非常。。。非常小。有句夸张的说法是：“在每秒钟产生一万亿个GUID的情况下，即使太阳变成白矮星的时候，它仍是唯一的”

GUID生成器的界面截图



GUID

- GUID的表示方法：

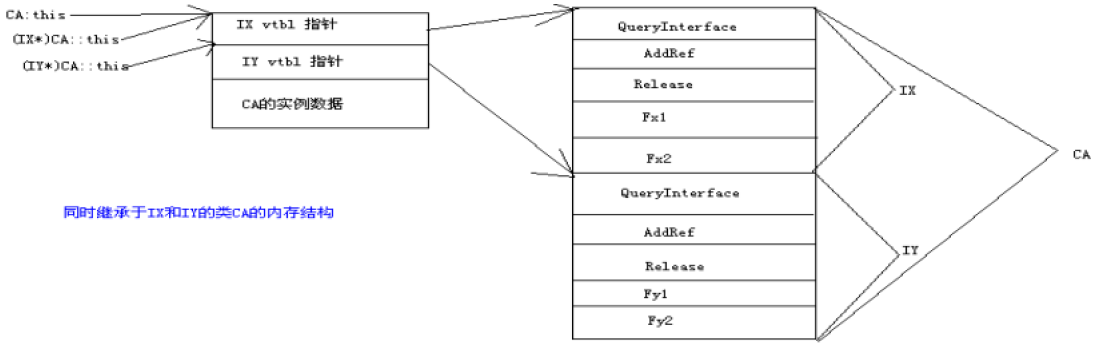
```
// {0E04C466-6CE9-4513-B306-43E8F7025EB9}

static const GUID guid =
{ 0xe04c466, 0x6ce9, 0x4513, { 0xb3, 0x6, 0x43, 0xe8, 0xf7, 0x2, 0x5e,
    0xb9 } };
```

QueryInterface

- 查看Section1Demo2例子，讲解QueryInterface的实现方法。

QueryInterface



同时继承于IX和IY的类CA的内存结构

回顾

- 这一讲，主要讲解如下要点：

一、现实中的组件与接口；

二、把现实中的思想融入到软件中；

三、c++程序中的组件与接口；

四、COM组件与COM接口；

五、QueryInterface函数，HRESULT类型，IID类型，数据类型转换。