实 验 十五 WMI 应用

15.1 实验目的

理解 WMI 工作原理，学习使用 WMI 技术获取机器信息，能够实现对机器的管理。

15.2 WMI 介绍

WMI（Windows 管理规范：Windows Management Instrumentation）是 Microsoft 基于 Web 的企业管理（WBEM）的实现，同时也是一种基于标准的系统管理接口。WMI 最早出现在 Microsoft Windows 2000 系统上，WMI 是一种轻松获取系统信息的强大工具。它的功能主要是：访问本地主机的一些硬件信息如内存，端口，磁盘驱动器；软件信息如服务，用户账号等。可使用 WMI 来查看硬盘空闲空间，使用 WMI 管理本地或远程计算机（当然你必须要拥有足够的权限），比如：重启，关机，关闭进程，创建进程，启动服务，它还可以订阅事件，实现对机器运行的监视功能。图15-1表示了 WMI 的体系结构。

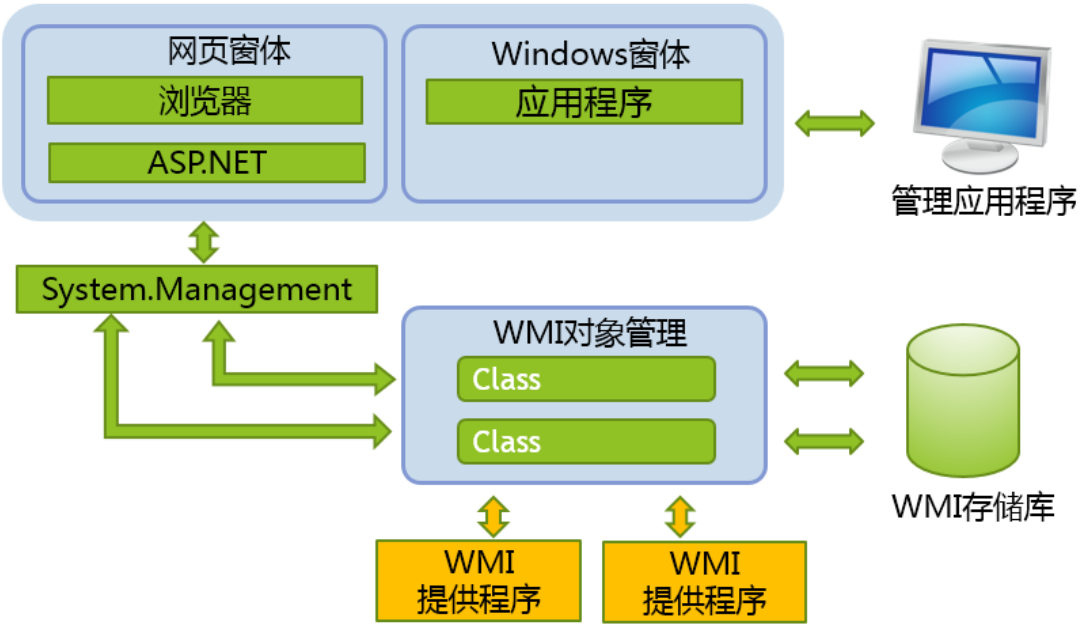

图 15-1 WMI 体系结构元素

WMI 操作使用查询语言，类似于 SQL 语言，这种语言叫做 WQL(WMI Query Language)，就是 SQL 操作，实际上是标准 SQL 的一个子集加上了 WMI 的扩展。例如创建一个要查询的

表 15-1 System.Management 命名空间中 WMI 类

类名	功能说明
ManagementObject	分别为单个管理对象或类。
ManagementClass	分别为单个管理对象或类。
ManagementObjectSearcher	用于根据指定的查询或枚举检索 ManagementObject 或 ManagementClass 对象的集合。
ManagementEventWatcher	用于预订来自 WMI 的事件通知。
ManagementQuery	所有查询类的基础。

对象可使用下面的语句：

New ManagementObjectSearcher("SELECT * FROM Win32_share")

表15-1说明了.net 类库里 System.Management 命名空间提供的类，图15-2表示了 WMI 类的继承关系。

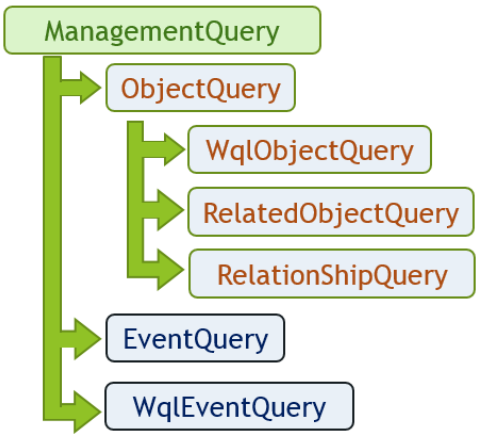

图 15-2 WMI 体系结构元素

在使用 WMI 时使用查询类和 WQL 查询语句的查询功能等效，比如下面两种方式得到的信息是相同的。

SelectQuery("Win32_LogicalDisk");

WqlObjectQuery("SELECT * FROM Win32_LogicalDisk");

使用 WMI 来查询和管理计算机可有下面的使用方案。

方案 1: 接收 WMI 数据

WMI 类提供有关计算机系统组件的信息，例如操作系统版本、硬盘驱动器上的可用空间、计算机的 IP 地址或有关计算机系统的各种其他数据。Win32 类（例如 Win32_LogicalDisk 和 Win32_OperatingSystem）是预装的 WMI 类，提供最常查询的信息。在应用程序中，可以通过查询所需的类属性来接收数据，可以使用 System.Management 命名空间中的类执行查询。

方案 2：执行 WMI 类中的方法

通过执行 WMI 类方法来控制不同计算机组件的行为。例如，可以执行 WMI 类 Win32_Process 的 Create 方法来启动进程，也可以执行 Win32_OperatingSystem 类的 Reboot 方法来重新启动计算机。可以使用 System.Management 命名空间中的类执行 WMI 类方法。

方案 3：从 WMI 接收事件

通过指定要接收的事件类型，创建用于接收由 WMI 引发的事件的客户端应用程序。例如，可以在每次启动或停止进程时接收事件，或在计算机上的注册表更改时接收事件，或者可接收到移动磁盘的加入或删除事件。

方案 4：访问远程计算机

可以使用 System.Management 命名空间中的类连接到远程计算机，机执行查询 WMI 数据、执行方法或接收事件。例如连接到远程计算机并检查远程计算机的硬盘驱动器上的可用磁盘空间，重新启动远程计算机，或在每次远程计算机上的注册表更改时接收事件通知。

方案 5：创建数据或事件提供程序

使用 System.Management.Instrumentation 命名空间中的类创建规范化的应用程序、WMI 数据或事件提供程序。为应用程序创建了数据或事件提供程序之后，其他应用程序可以发现包含提供程序的应用程序，监视您在数据提供程序中定义的应用程序属性，并使用 WMI 配置对象，以获取您提供的数据。允许其他用户和应用程序使用 WMI 查询与您的应用程序有关的数据。

15.3 使用 WMI 操作机器

本实验使用 WMI 实现获取机器信息和对 U 盘插入机器的监视，参考的程序界面如图15-3所示。程序需要在.NET 标签页添加 System.Managment 引用，在源代码文件中添加 System.Managment 命名空间。

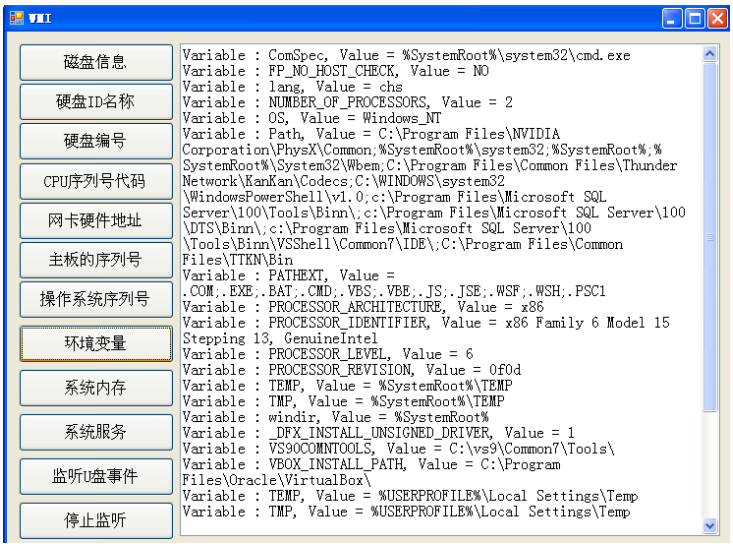

图 15-3 使用 WMI 获得机器信息

15.3.1 WMI 获得机器信息

实现获取硬盘信息的代码片段如下：

```
SelectQuery query = new SelectQuery("Select * From Win32_LogicalDisk");
ManagementObjectSearcher searcher = new ManagementObjectSearcher(query);
textBox1.Text = "";
//1 No type
//2 Floppy disk
```

```
//3 Hard disk
//4 Removable drive or network drive
//5 CD-ROM
//6 RAM disk
foreach (ManagementBaseObject disk in searcher.Get())
{
  textBox1.Text+=disk["Name"] + " " + disk["DriveType"] + " " +
disk["VolumeName"]+"\r\n";
}
```

获取硬盘编号的代码片段如下：

```
ManagementObjectSearcher searcher = new ManagementObjectSearcher("SELECT * FROM
Win32_PhysicalMedia");
ManagementObjectCollection moCollection = searcher.Get();
textBox1.Text = "";
foreach (ManagementObject mObject in moCollection)
{
  textBox1.Text += mObject["SerialNumber"].ToString() + " ";
}
```

获取网卡硬件地址代码如下：

```
ManagementClass mc = new ManagementClass("Win32_NetworkAdapterConfiguration");
ManagementObjectCollection moc = mc.GetInstances();
textBox1.Text = "";
foreach(ManagementObject mo in moc)
{
  if((bool)mo["IPEnabled"] == true)
  textBox1.Text += string.Format("MAC address {0}", mo["MacAddress"].ToString()) +
   "\r\n";
  mo.Dispose();
}
```

获取系统服务程序列表：

```
ManagementObjectSearcher searcher = new ManagementObjectSearcher("SELECT * FROM
Win32_Service");
ManagementObjectCollection moCollection = searcher.Get();
textBox1.Text = "";
foreach (ManagementObject mo in moCollection)
{
  string state;
  if (mo["Started"].Equals(true))
    state = "Started";
  else
```

```
        state = "Stop";
    textBox1.AppendText(string.Format(" 服务名：{0}，启动方式：{1}，
    状态：{2}，启动账号：{3}\r\n",
        mo["Name"].ToString(), mo["StartMode"].ToString(), state, mo["StartName"].ToString()));
  }
```

15.3.2 使用 WMI 订阅 U 盘插入事件

可利 WMI 对机器的特定事件进行订阅，这里对用户 U 盘插入机器的事件进行监视。首先编写工作线程代码：

```
  static void uWatch()
  {
  //TargetInstance.DriveType = 2
  //代表判断 Win32_LogicalDisk.DriveType 属性，2 则代表可移动磁盘
  WqlEventQuery queryCreate = new WqlEventQuery("__InstanceCreationEvent",
    new TimeSpan(0, 0, 1),
  "TargetInstance ISA \"Win32_LogicalDisk\"");
  // AND TargetInstance.DriveType = 2
  ManagementEventWatcher watcher =
    new ManagementEventWatcher(queryCreate);
  //watcher.Options.Timeout = new TimeSpan(0, 0, 35);
  ManagementBaseObject ex = watcher.WaitForNextEvent();
  //textBox1.Text = ((ManagementBaseObject)ex["TargetInstance"])["Name"].ToString();
  MessageBox.Show("U 盘插入");
  watcher.EventArrived += new EventArrivedEventHandler(HandleEvent);
  watcher.Start();
  uwatch_e.WaitOne(60000);
  watcher.Stop();
  }
```

启动线程的事件函数如下：

```
  static ManualResetEvent uwatch_e;
  private void button11_Click(object sender, EventArgs e)
  {
    uwatch_e = new ManualResetEvent(false);
    ThreadStart ts = new ThreadStart(uWatch);
    Thread th = new Thread(ts);
    th.IsBackground = true;
    th.Start();
  }
```

异步回调函数的定义：

```
static private void HandleEvent(object sender,EventArrivedEventArgs e)
{
  MessageBox.Show("U 盘插入");
  uwatch_e.Set();
}
```

编译运行程序，当用户插入 U 盘时，程序可收到事件并弹出消息窗体。

15.3.3 使用 WMI 获得进程创建信息

15.4 实验作业

1. 利用 WMI 启动一个 explorer.exe 进程。
2. 利用 WMI 设置本地连接的 IP 地址。
3. 利用 WMI 实现对应用进程创建的监视。