Schema Design

The MappingStrings Qualifier

If you are exposing your own management objects through a WMI provider, then knowing about the `MappingStrings` qualifier will be of interest to you. The `MappingStrings` qualifier describes the origin of properties. This is both useful to you, the schema designer, and also to the user of your schema. It helps you document where the values for properties originate which may become useful during the implementation of your WMI provider. It may also be useful to the users of your class schema to help them understand where property values originate. In some cases, this information may be useful when diagnosing problems where the original source of the information can be directly inspected.

The data type of the `MappingStrings` qualifier is an array of strings and the format of the strings can take any form. As you will see in a moment, Microsoft have adopted to use a '|' delimiter to form a path to the location of the property.

Let's have a look at a trimmed version of the `Win32_Service` class:

```
[dynamic, provider("CIMWin32")]
class Win32_Service : Win32_BaseService
{
    [ read,
      MappingStrings{"Win32API|Service Structures|SERVICE_STATUS|dwCheckPoint"}
    ]
    uint32 CheckPoint;

    [ read,
      MappingStrings{"Win32API|Service Structures|SERVICE_STATUS|dwWaitHint"}
    ] uint32 WaitHint;

    [ read,
      MappingStrings{"Win32API|Service Structures|SERVICE_STATUS_PROCESS|dwProcessId"}
    ] uint32 ProcessId;
};
```

From a developer point of view, it should be straight forward to understand where the property values originate. The value of the `CheckPoint` property originates from a `dwCheckPoint` member in the Windows service data structure, `SERVICE_STATUS`, which is part of the Win32 API. Notice how the location is specified using the '|' delimiter. We recommend that you use a similar mechanism in your own schemas.

The `SERVICE_STATUS` structure is used by `ControlService`, `EnumDependentServices`, `EnumServicesStatus`, and `QueryServiceStatus` APIs. So we know that the management objects and properties were created based on the information from one of these APIs. For your information, the `SERVICE_STATUS` structure looks like this:

```
typedef struct _SERVICE_STATUS
{
    DWORD dwServiceType;
    DWORD dwCurrentState;
    DWORD dwControlsAccepted;
    DWORD dwWin32ExitCode;
    DWORD dwServiceSpecificExitCode;
```

```
            DWORD dwCheckPoint;
            DWORD dwWaitHint;
        } SERVICE_STATUS, *LPSERVICE_STATUS;
```

Let's have a look at a class that we defined in Chapter 12. The `Sample_MyUser` class illustrated that property values can originate from different locations. The main purpose of the example was to demonstrate how properties from different sources can be logically encapsulated into one class (or more appropriately, a management object).

```
    class Sample_MyUser
    {
        [key] string Name;              // Stored in a database
        boolean OnLine;                 // Retrieved from the Internet
        boolean UserAlreadyLoggedOn;    // Obtained at runtime
        string PhoneNumber;             // Retrieved from the Active Directory
        Sample_Preferences Options;     // Stored in an XML file
        boolean EnableUser;             // Stored in Registry
    };
```

Let's extend the class definition by documenting where the class's properties would originate (probably if this were a real management class).

```
    class Sample_MyUser
    {
        [ key,
          MappingStrings{"ADO|DSN|Provider=Microsoft.Jet.OLEDB.4.0;
            Data Source=Users.mdb|Table=RegUsers|Column=UserName"}
        ]
        string Name;

        [MappingStrings{"Internet|Microsoft|Passport"}]
        boolean OnLine;

        [MappingStrings{"MemoryMappedFile|LoggedOnUsers"}]
        boolean UserAlreadyLoggedOn;

        [MappingStrings{"ADSI|LDAP://CN=User,CN=Schema,CN=Configuration,
           DC=thebighouse,DC=com|telephoneNumber"}]
        string PhoneNumber;

        [MappingStrings{"XML|userspref.xml"}]
        Sample_Preferences Options;

        [MappingStrings{"HKLM|Software\MyApp\Users\UserEnabled"}]
        boolean EnableUser;
    };
```

The `Name` property originates from a database which is specified using an ADO (Active Data Objects) DSN (Data Source Name). The `OnLine` property is determined through Microsoft Passport. `UserAlreadyLoggedOn` is determined using a memory mapped file called `LoggedOnUsers`. `PhoneNumber` is retrieved from the Active Directory – specifically, from the `telephoneNumber` attribute in the `Users` directory class. The user's `Options` is obtained from an XML file called `userspref.xml`. `EnableUser` is configured in the registry.

As you can see, documenting the origin of properties is straightforward. Remember that the `MappingStrings` qualifier uses an array of strings data type – this means that you can specify the origin the property (or management object) using several strings. In the previous examples, we have used just one string to describe the origin.