How to get operating system version information (part 4 of 6)

OS information stored in the registry

Getting Information About NT 4 from the Registry

It has been noted above that the extended operation system information provided by the Windows API is not available in versions of NT 4 prior to Service Pack 6. In this section we will look at how to overcome this problem by reading the required information from the registry.

There are two pieces of relevant information we can get from the registry:

- The NT product type.
- Whether NT 4 Service Pack 6 or 6a is installed.

Product Type

Let us first look at how to get the product type. Windows stores information about the edition of the NT4 OS in the registry under this key:

```
HKLM\SYSTEM\CurrentControlSet\Control\ProductOptions
```

The key's *ProductName* value stores a code that indicates the product type. Values and their interpretation are shown in *Table 6*.

NT4 Registry Entries	
Code	**Edition**
`WINNT`	Workstation
`LANMANNT`	Server
`SERVERNT`	Advanced Server

Table 6

Service Pack 6a

We detect whether NT Service Pack 6a is installed by checking that *Win32CSDVersion* contains the string 'Service Pack 6' and then checking for the existence of the registry key:

```
HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Hotfix\Q246009
```

If the key exists we have Service Pack 6a.

OS Version Information Class: Version 3

Having seen what information we can find from the registry, and how to get it, we can update our operating system information class to work correctly with earlier versions of Windows NT 4.

First we will define a couple of new private static methods to get the required information from the registry and will then update various public methods.

Registry Access Methods

We will create two new private methods that access the registry: *GetNT4ProductType* to read the product type and *IsNT4SP6a* to check for NT 4 Service Pack 6a.

GetNT4ProductType will read the product type from the registry and return the equivalent *TOSProductType* code. Before doing this we will need to define a new *TOSProductType* value – *ptNTAdvancedServer* to represent NT 4 Advanced Server. *Listing 21* shows the revised declaration of *TOSProductType*.

```
type
  TOSProductType = (
    ptNA,                 // not applicable: not a Windows NT platform
    ptUnknown,            // unknown NT product type
    ptNTWorkstation,      // NT workstation
    ptNTServer,           // NT server
    ptNTDomainController, // NT domain controller
    ptNTAdvancedServer    // NT advanced server
  );
```

Listing 21

Having extended *TOSProductType* we can now implement *GetNT4ProductType* as per *Listing 22*.

```
class function TOSInfo.GetNT4ProductType: TOSProductType;
var
  Reg: TRegistry;
  ProductType: string;
begin
  Result := ptUnknown;
  Reg := TRegistry.Create;
  try
    Reg.RootKey := HKEY_LOCAL_MACHINE;
    if Reg.OpenKeyReadOnly(
      'SYSTEM\CurrentControlSet\Control\ProductOptions'
    ) then
    begin
      ProductType := Reg.ReadString('ProductType');
      if SameText(ProductType, 'WINNT') then
        Result := ptNTWorkstation
      else if SameText(ProductType, 'LANMANNT') then
        Result := ptNTServer
      else if SameText(ProductType, 'SERVERNT') then
        Result := ptNTAdvancedServer;
      Reg.CloseKey;
    end;
  finally
    Reg.Free;
  end;
end;
```

Listing 22

We first set a default result of *ptUnknown* in case we can't read an expected value from the registry. Then we create a *TRegistry* object via which we access the registry. Next we open the key described in the previous section and read the *ProductType* value. Finally the product type code is mapped onto an equivalent *TOSProductType* value. Remember that *Table 6* lists the possible values of the *ProductType* registry entry.

Now we can look at the definition of *IsNT4SP6a*. See *Listing 23*.

```
class function TOSInfo.IsNT4SP6a: Boolean;
var
  Reg: TRegistry;
begin
  if (Product = osWinNT4)
    and SameText(Win32CSDVersion, 'Service Pack 6') then
  begin
    // System is reporting NT4 SP6
    // we have SP 6a if particular registry key exists
    Reg := TRegistry.Create;
    try
      Reg.RootKey := HKEY_LOCAL_MACHINE;
      Result := Reg.KeyExists(
        'SOFTWARE\Microsoft\Windows NT\CurrentVersion\Hotfix\Q246009'
      );
    finally
```

```
      Reg.Free;
    end;
  end
  else
    // System not reporting NT4 SP6, so not SP6a!
    Result := False;
end;
```

Listing 23

First we check whether we have Windows NT 4 and that Service Pack 6 is being reported by Delphi's *Win32CSDVersion* global variable. Then we create a *TRegistry* object and use it to check for the presence of the registry key that indicates Service Pack 6a is present. We return true if the key is present and false otherwise.

Now we have the registry access methods we need it's time to move on to modify some of *TOSInfo's* public methods to make use of the registry information.

Updated Public Methods

We need to update four public methods to take account of the new information we have about NT 4. They are:

- *ProductType* – to get the NT 4 product types from the registry.
- *IsServer* – to take account of the new *ptNTAdvancedServer* product type.
- *Edition* – to describe the NT 4 editions that we learn about from the registry.
- *ServicePack* – to include detection of NT 4 service pack 6a.

We begin with *ProductType*. *Listing 24* shows the revised method in full.

```
class function TOSInfo.ProductType: TOSProductType;
begin
  if IsWinNT then
  begin
    if Win32HaveExInfo then
    begin
      case Win32ProductType of
        VER_NT_WORKSTATION: Result := ptNTWorkstation;
        VER_NT_SERVER: Result := ptNTServer;
        VER_NT_DOMAIN_CONTROLLER: Result := ptNTDomainController;
        else Result := ptUnknown;
      end;
    end
    else
      Result := GetNT4ProductType;
  end
  else
    Result := ptNA;
end;
```

Listing 24

We now check if *Win32HaveExInfo* is true and proceed as before if so. When extended version information is not present we know we must look in the registry for the product type, which we do by calling the new *GetNT4ProductType* private method.

Next up is *IsServer*. As *Listing 25* shows, all we need to do is add *ptNTAdvancedServer* to the list of server product types that are tested.

```
class function TOSInfo.IsServer: Boolean;
begin
  Result := ProductType in
    [ptNTServer, ptNTDomainController, ptNTAdvancedServer];
end;
```

Listing 25

The changes we must make to the *Edition* method are show in *Listing 26*. For the sake of brevity the listing shows only the required changes and the surrounding context.

```
class function TOSInfo.Edition: string;
begin
  Result := '';
  if IsWinNT then
  begin
    if Win32HaveExInfo then
    begin
      ...
    end
    else
    begin
      // No extended info: use regsitry
      case GetNT4ProductType of
        ptNTWorkstation: Result := 'WorkStation';
        ptNTServer: Result := 'Server';
        ptNTAdvancedServer: Result := 'Advanced Server'
      end;
      if Result <> '' then
        Result := Result + Format(
          ' %d.%d', [Win32MajorVersion, Win32MinorVersion]
        );
    end;
  end;
end;
```

Listing 26

Taking our lead from *Listing 24*, we add an else clause that executes when *Win32HaveExInfo* returns false on an NT platform OS. The code in the else clause gets the product type from the registry by calling *GetNT4ProductType* and then returns a description of the returned product type. We then append the major and minor versions of the OS to the description.

The last method to be modified is *ServicePack*. *Listing 27* shows the changes. Only the code relating to the NT platform is changed, so the existing Windows 9x code is omitted from the listing.

```
class function TOSInfo.ServicePack: string;
begin
  Result := '';
  if IsWin9x then
  begin
    ...
  end
  else if IsWinNT then
  begin
    if IsNT4SP6a then
      Result := 'Service Pack 6a'
    else
      Result := Win32CSDVersion;
  end;
end;
```

Listing 27

All that needs to be done after testing for the Windows NT platform is to call *IsNT4SP6a* to check for NT 4's service pack 6a. If SP6a is found, we return 'Service Pack 6a', otherwise the value of the *Win32CSDVersion* variable is returned as before.

We have now completed our review of NT 4 version information that can be read from the registry. In the *next section* we will find out more about different versions of Windows XP.
