

Re: Reading names of comports [COM1, COM2 etc]

Source: <http://coding.derkeiler.com/Archive/Delphi/alt.comp.lang.borland-delphi/2005-08/msg00089.html>

- *From:* Ekkehard Domning <edo@xxxxxxxx>
 - *Date:* Sun, 07 Aug 2005 14:36:57 +0200
-

Hello Vic,

Vic Fraenckel schrieb:

I am attempting to read the names of the known com ports on my computer using the following function.

as You see, You dived into a "not simple" problem.

First:

Dealing with the StringList and the Registry is described in the other postings, so I leave You alone with this :-)

Second:

While all Comports are in the "HARDWARE\DEVICEMAP\SERIALCOMM" key under NT/2K/XP they are under a different location under 9x/ME. Harder, under 9x/ME, finally a lot of Keys must be scanned to find virtual Modems creating there own commports.

There might be also the possibility that a commport exists, but is not in the registry. I have never seen this behaviour but I have heard about. Probably than is seems to be usefull to probe commport names like "COM1"... if they exists. The problem there is that the createfile fill fail either on non existng ports or if the port is allready in use.

I use the function below, (could also found in my OpenSource SerialNG component <http://www.domis.de/serialng.htm> in the file "CommPortList.pas"), which scans the registry only and do no probing. I found that this works well on all systems, that I have used so far.

Best regards
Ekkehard Domning

```
procedure GetPortList(Strings : TStrings);  
var Reg : TRegistry;
```

Re: Reading names of comports [COM1, COM2 etc]

```
procedure ScanRegHardware;
var
  i : integer;
  PortName : string;
  LName : TStringList;
begin
  if Reg.OpenKeyReadOnly('\hardware\devicemap\serialcomm') then
  begin
    LName := TStringList.Create;
    Reg.GetValueNames(LName);
    for i := 0 to LName.Count - 1 do
    begin
      if Reg.GetDataTypes(LName.Strings[i]) = rdString then
      begin
        PortName := Reg.ReadString(LName.Strings[i]);
        if Strings.IndexOf(PortName) < 0 then
          Strings.Add(PortName);
        end
      end;
    LName.Free;
  end
end;

procedure ScanRegEnum(Key : String);
// This Subprocedure recurses thru all keys below key
var
  LKey : TStringList;
  LName : TStringList;
  i : Integer;
  Driver, PortName : String;
  PortSubClass : Byte;
begin
  if not Reg.OpenKeyReadOnly(Key) then // Danke Andreas Schmidt!
  Exit;
  LName := TStringList.Create;
  Reg.GetValueNames(LName);
  i := LName.IndexOf('class');
  if i >= 0 then
  begin
    if (Reg.GetDataTypes('class') = rdString) and
    ((LowerCase(Reg.ReadString('class')) = 'ports') or //normal Serialports
    like COMx
    (LowerCase(Reg.ReadString('class')) = 'modem')) and //some abnormal
    onboard Modems
    (Reg.GetDataTypes('driver') = rdString) and
    (Reg.GetDataTypes('portname') = rdString) then
    begin
      Driver := Reg.ReadString('driver');
      PortName := Reg.ReadString('portname');
      if Reg.OpenKeyReadOnly('\System\CurrentControlSet\Services\Class\' +
      Driver) and
      (Reg.ReadBinaryData('PortSubClass',PortSubClass,1) = 1) and
      ((PortSubClass = 1) or //Ports
      (PortSubClass = 2)) and //Modems
```

Re: Reading names of comports [COM1, COM2 etc]

```
(Strings.IndexOf(PortName) < 0) then
Strings.Add(PortName);
end
end;
LName.Free;
Reg.OpenKeyReadOnly(Key);
if Reg.HasSubKeys then
begin
LKey := TStringList.Create;
Reg.GetKeyNames(LKey);
for i := 0 to LKey.Count - 1 do
begin
ScanRegEnum(Key + '\' + LKey[i]);
end;
LKey.Free;
end;
end;

var
  VersionInfo : TOSVersionInfo;
begin
  if CheckOS(VersionInfo) > 0 then
    if VersionInfo.dwMajorVersion >= 4 then
      begin
        Reg := TRegistry.Create;
        Reg.RootKey := HKEY_LOCAL_MACHINE;
        if VersionInfo.dwMajorVersion > 4 then
          ScanRegHardware //Win 2k, XP
        else if VersionInfo.dwMinorVersion > 0 then
          ScanRegEnum('\enum') //Win9x, mE
        else
          ScanRegHardware; //WinNT
        Reg.Free;
      end;
    end;
end;
```

the code uses also a function for OS determination

```
function CheckOS(var VersionInfo : TOSVersionInfo) : Integer;
begin
  {
  dwOSVersionInfoSize: DWORD;
  dwMajorVersion: DWORD;
  dwMinorVersion: DWORD;
  dwBuildNumber: DWORD;
  dwPlatformId: DWORD;
  szCSDVersion: array[0..127] of AnsiChar; // Maintenance string for PSS
  usage
  }
  VersionInfo.dwOSVersionInfoSize := SizeOf(VersionInfo);
```

Re: Reading names of comports [COM1, COM2 etc]

```
if GetVersionEx(VersionInfo) then
CheckOS := VersionInfo.dwPlatformId
else
CheckOS := -1;
end;
.
```