

Forum:[Hack Forums](#)

Thread: [netshare wapper references](#)

Luuna - January 21, 2022, 9:13 am

Hi all, just wondering if anyone has any resources to any netshare wrappers, im writing my own but wanted to see if there was any existing.

```
Code public static string getmappeddrive_UNC()
{
    string a = "";
    var searcher = new ManagementObjectSearcher(
        "root\\CIMV2",
        "SELECT * FROM Win32_MappedLogicalDisk");
    try
    {
        foreach (ManagementObject queryObj in searcher.Get())
        {
            a = queryObj["ProviderName"].ToString();
        }
    }
    catch
    {
    }
    return a;
}
```

I plan to move to Pvoke but i was just seeing if anyone else had any remove netshare wrappers would be pinvoke be better?

<https://ehikioya.com/get-network-path-mapped-drive/>

would be pinvoke be better?

<https://ehikioya.com/get-network-path-mapped-drive/>

Never mind found what i was looking for. reference to other if ever needed.

```
[DllImport("mpr.dll", CharSet = CharSet.Unicode, SetLastError = true)]
public static extern int WNetGetConnection(
    [MarshalAs(UnmanagedType.LPTStr)] string localName,
    [MarshalAs(UnmanagedType.LPTStr)] StringBuilder remoteName,
    ref int length);
```

```
public static string GetUNCPath(string originalPath)
{
    StringBuilder sb = new StringBuilder(512);
    int size = sb.Capacity;

    if (originalPath.Length > 2 && originalPath[1] == ':')
    {
        char c = originalPath[0];
```

```

        if ((c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z'))
        {
            int error = WNetGetConnection(originalPath.Substring(0, 2), sb, ref size);
            if (error == 0)
            {
                DirectoryInfo dir = new DirectoryInfo(originalPath);
                string path =
Path.GetFullPath(originalPath).Substring(Path.GetPathRoot(originalPath).Length);
                return Path.Combine(sb.ToString().TrimEnd(), path);
            }
        }
    }

    return originalPath;
}

```

Never mind found what i was looking for. reference to other if ever needed.

```

[DllImport("mpr.dll", CharSet = CharSet.Unicode, SetLastError = true)]
public static extern int WNetGetConnection(
[MarshalAs(UnmanagedType.LPTStr)] string localName,
[MarshalAs(UnmanagedType.LPTStr)] StringBuilder remoteName,
ref int length);

public static string GetUNCPath(string originalPath)
{
    StringBuilder sb = new StringBuilder(512);
    int size = sb.Capacity;

    if (originalPath.Length > 2 && originalPath[1] == ':')
    {
        char c = originalPath[0];
        if ((c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z'))
        {
            int error = WNetGetConnection(originalPath.Substring(0, 2), sb, ref size);
            if (error == 0)
            {
                DirectoryInfo dir = new DirectoryInfo(originalPath);
                string path =
Path.GetFullPath(originalPath).Substring(Path.GetPathRoot(originalPath).Length);
                return Path.Combine(sb.ToString().TrimEnd(), path);
            }
        }
    }

    return originalPath;
}

```

Never mind found what i was looking for. reference to other if ever needed.

```

[DllImport("mpr.dll", CharSet = CharSet.Unicode, SetLastError = true)]
public static extern int WNetGetConnection(
[MarshalAs(UnmanagedType.LPTStr)] string localName,
[MarshalAs(UnmanagedType.LPTStr)] StringBuilder remoteName,

```

```
ref int length);
```

```
public static string GetUNCPath(string originalPath)
{
    StringBuilder sb = new StringBuilder(512);
    int size = sb.Capacity;

    if (originalPath.Length > 2 && originalPath[1] == ':')
    {
        char c = originalPath[0];
        if ((c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z'))
        {
            int error = WNetGetConnection(originalPath.Substring(0, 2), sb, ref size);
            if (error == 0)
            {
                DirectoryInfo dir = new DirectoryInfo(originalPath);
                string path =
Path.GetFullPath(originalPath).Substring(Path.GetPathRoot(originalPath).Length);
                return Path.Combine(sb.ToString().TrimEnd(), path);
            }
        }
    }

    return originalPath;
}
```