

# Mikael Chudinov. My personal blog

from “if it ain’t broke, don’t fix it” to “if it ain’t perfect, keep improving”

## Packaging of a .NET application on Windows

This post is about how to package .NET application for Windows into an MSI (**Micro**Soft **I**nstaller) package using Wix# tool in command line. Packaging process can be easily automated on continues integration server.



Generally application building workflow is simple:

- Assign a version number to an assembly
- Build binaries
- Copy binaries to package folder and build msi-package with the same version as assembly

Build process will be controlled by MSBuild project file and can be started from a continues integrations server.

Sample solution is available for download here <https://github.com/mchudinov/PackagingMSI>. Solution is compatible with Visual Studio 2013.

Automated building and versioning processes were described in my previous posts:

- [Building and Testing](#)
- [Versioning](#)

I need to create a MSI Windows package from my .NET binaries. The result should be a package that

does the following while installation:

- Package version is the same as an assembly version of an application
- Shows standard Windows graphical installation wizard
- Installs binary by default into a predefined folder. It is typically %Program Files% folder.

I use [Wix#](#) that automates WiX toolset that automates creation of MSI packages. [Wix#](#) is a framework for building a complete MSI or WiX source code by using script files written with the C# syntax.

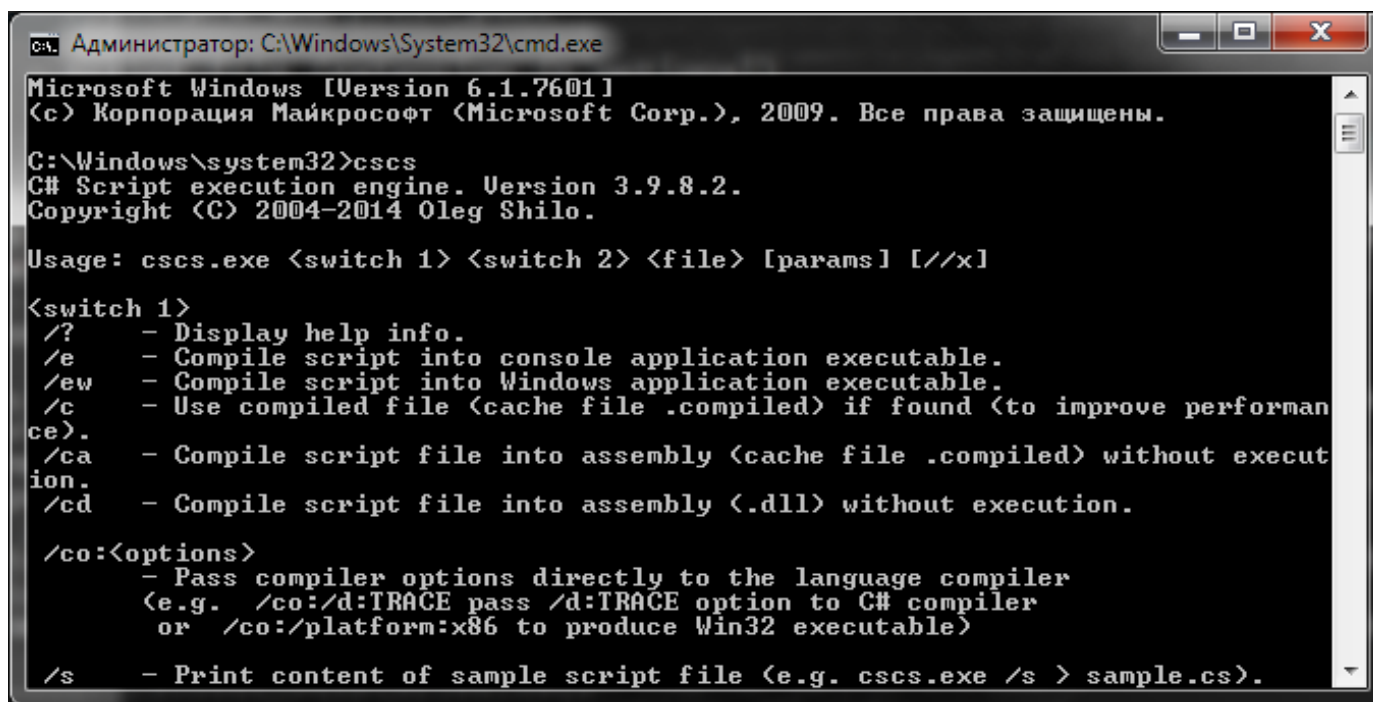
## 1. Prepare packaging environment

Since Wix# script is written in C# and will run in command line I need a C# scripting engine. I use [CS-Script](#) engine. CS-Script must be installed on the continuous integration server and added to PATH in order to build packages there.

Installation process is simple:

1. Download the CS-Script archive from [official website](#)
2. Unarchive it to a folder without spaces. I use c:\lib\cs-script
3. Add this folder to the PATH environment variable
4. Follow instructions in readme.txt

Check that CS-Script engine works after installation and it was added to PATH. Run **cscs** command in command line:



```
Администратор: C:\Windows\System32\cmd.exe
Microsoft Windows [Version 6.1.7601]
(c) Корпорация Майкрософт (Microsoft Corp.), 2009. Все права защищены.

C:\Windows\system32>cscs
C# Script execution engine. Version 3.9.8.2.
Copyright (C) 2004-2014 Oleg Shilo.

Usage: cscs.exe <switch 1> <switch 2> <file> [params] [/x]

<switch 1>
/?      - Display help info.
/e      - Compile script into console application executable.
/ew     - Compile script into Windows application executable.
/c      - Use compiled file <cache file .compiled> if found <to improve performance>.
/ca     - Compile script file into assembly <cache file .compiled> without execution.
/cd     - Compile script file into assembly <.dll> without execution.

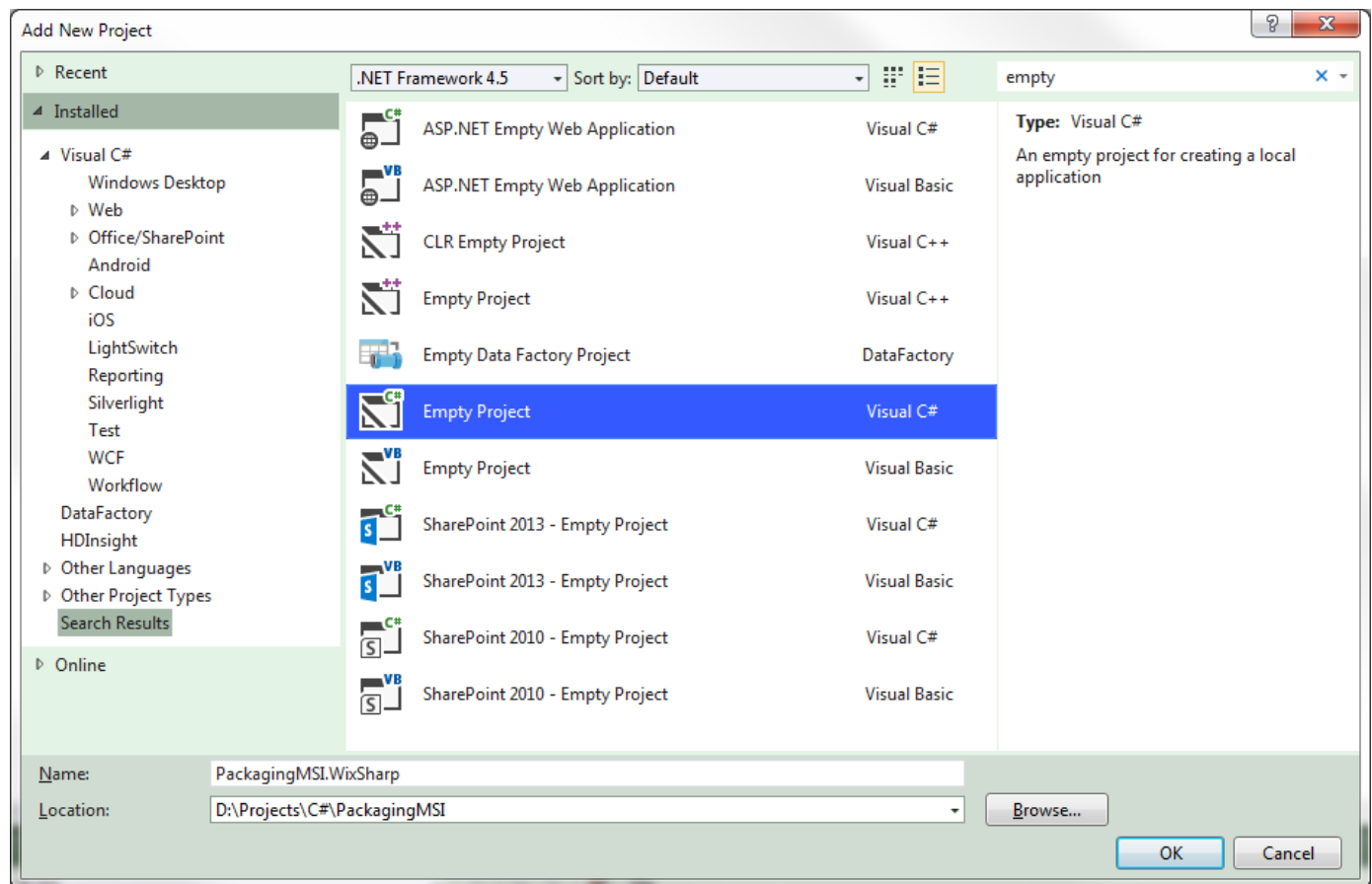
/co:<options>
        - Pass compiler options directly to the language compiler
        <e.g. /co:/d:TRACE pass /d:TRACE option to C# compiler
          or /co:/platform:x86 to produce Win32 executable>

/s      - Print content of sample script file <e.g. cscs.exe /s > sample.cs>.
```

## 2. Create packaging project

### 2.1 Add an empty C# project to the solution.

Packaging script should be added to a separate project. I name it <MyProjectName>.WixSharp. It should be an empty C# project.



### 2.2 Add [WixSharp NuGet package](#) to this project.

### 2.3 Add a public class **Script** in this new project.

**Script.cs** must have the method **static public void Main(string[] args)**. This will be my packaging script. Add following using to the file:

```
using System;
using System.Diagnostics;
//css_ref %WIXSHARP_DIR%\wixsharp.dll;
using WixSharp;
```

Note `//css_ref %WIXSHARP_DIR%\wixsharp.dll` is commented out. This is how it must be. This is a

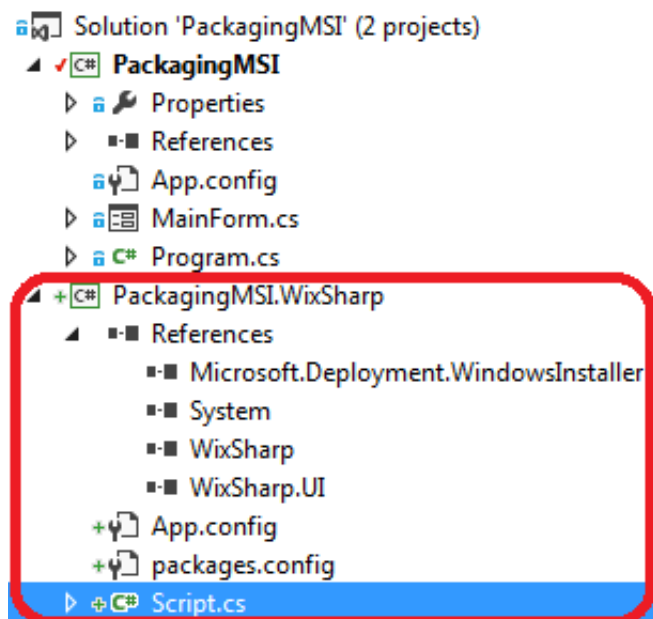
directive for cs-script.

Keep method Main simple for a while:

```
using System;
using System.Diagnostics;
//css_ref %WIXSHARP_DIR%\wixsharp.dll;
using WixSharp;

public class Script
{
    static public void Main(string[] args)
    {
        Console.WriteLine("Hello World!");
    }
}
```

The new project should look like this:

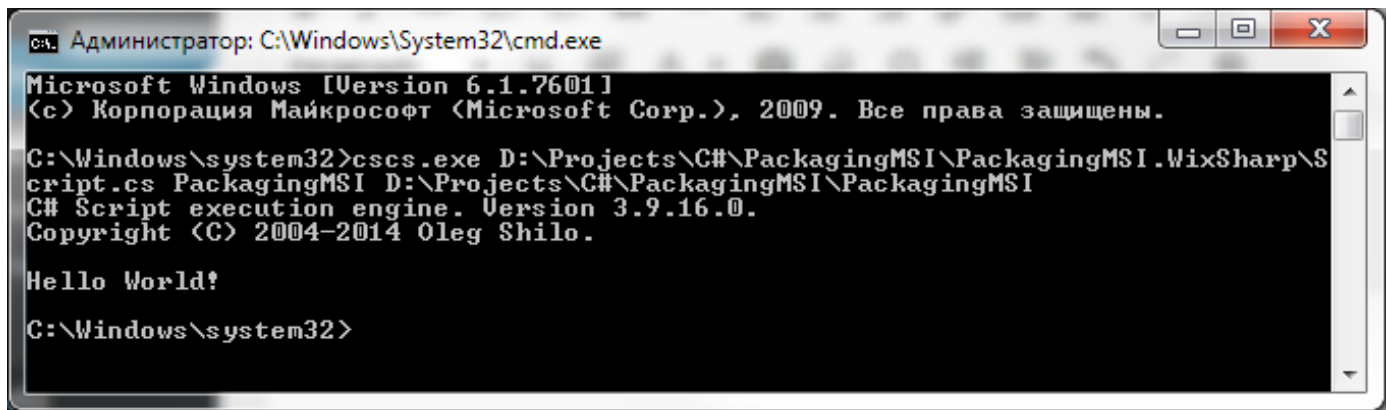


## 2.4 Test that CS-Script engine can run Script.cs

Write the following command in Windows command line:

```
cscs.exe <full path to project>\<project name>.WixSharp\Script.cs <project name> <full path
```

You should see a Hello World! response from CS-Script running Script.cs



```
Администратор: C:\Windows\System32\cmd.exe
Microsoft Windows [Version 6.1.7601]
(c) Корпорация Майкрософт (Microsoft Corp.), 2009. Все права защищены.

C:\Windows\system32>cscs.exe D:\Projects\C#\PackagingMSI\PackagingMSI.WixSharp\S
cript.cs PackagingMSI D:\Projects\C#\PackagingMSI\PackagingMSI
C# Script execution engine. Version 3.9.16.0.
Copyright (C) 2004-2014 Oleg Shilo.

Hello World!

C:\Windows\system32>
```

### 3. Create building script for the solution

Add simple MSBuild script to the project. Read more about MSBuild in my [Building and Testing](#) blogpost. This script will build and package application.

Build steps are the following:

- Clean solution
- Restore NuGet packages
- Set version to assemblies
- Compile project

Note that versioning needs [MSBuild.Extension.Pack](#) and [MSBuildTasks](#) NuGet packages be installed.

```
<?xml version="1.0" encoding="utf-8"?>
<Project xmlns="http://schemas.microsoft.com/developer/msbuild/2003" DefaultTargets="Run">

  <PropertyGroup>
    <Configuration>Release</Configuration>
    <SolutionName>PackagingMSI</SolutionName>
  </PropertyGroup>

  <Target Name="Run">
    <CallTarget Targets="Clean" />
    <CallTarget Targets="Restore" />
    <CallTarget Targets="Version" />
    <CallTarget Targets="Build" />
  </Target>

  <Target Name="Clean">
    <Message Text="Clean" />
    <ItemGroup>
      <FilesToDeleteInPackageFolder Include="*.msi"/>
      <FilesToDeleteInPackageFolder Include="$(SolutionName).WixSharp/*.msi"/>
    </ItemGroup>
    <Delete Files="@ (FilesToDeleteInPackageFolder) " />
  </Target>
</Project>
```

```

</Target>

<Target Name="Restore">
  <Message Text="Restore NuGet packages" />
  <Exec Command="nuget.exe restore" ContinueOnError="False"/>
</Target>

<UsingTask AssemblyFile="packages/MSBuild.Extension.Pack.1.6.0/tools/net40/MSBuild.Exte
<Target Name="Version">
  <Message Text="Versioning assemblies" />

  <ItemGroup>
    <AssemblyInfoFiles Include="**\AssemblyInfo.cs" />
  </ItemGroup>

  <AssemblyInfo
    AssemblyInfoFiles="@ (AssemblyInfoFiles) "

    AssemblyMajorVersion="$(MajorVersion) "
    AssemblyMinorVersion="$(MinorVersion) "
    AssemblyBuildNumberType="DateString"
    AssemblyBuildNumberFormat="MMdd"
    AssemblyRevisionType="AutoIncrement"
    AssemblyRevisionFormat="000"

    AssemblyFileMajorVersion="$(MajorVersion) "
    AssemblyFileMinorVersion="$(MinorVersion) "
    AssemblyFileBuildNumberType="DateString"
    AssemblyFileBuildNumberFormat="MMdd"
    AssemblyFileRevisionType="AutoIncrement"
    AssemblyFileRevisionFormat="000"

  />
</Target>

<Target Name="Build">
  <Message Text="Build $(Configuration)" />
  <MSBuild Projects="$(SolutionName)/$(SolutionName).csproj" Properties="Configuration=
</Target>

<Target Name="Pack">
  <Message Text="Pack application into MSI" />
  <ItemGroup>
    <FilesToDeleteInPackageFolder Include="*.msi"/>
    <FilesToDeleteInPackageFolder Include="$(SolutionName).WixSharp/*.msi"/>
  </ItemGroup>
  <Delete Files="@ (FilesToDeleteInPackageFolder)"/>
  <Exec Command="cscs.exe $(SolutionName).WixSharp/Script.cs $(SolutionName) $(MSBuild
</Target>

</Project>

```

Test build in Visual Studio command line **msbuild Build.proj**:

```

Администратор: Developer Command Prompt for VS2013
D:\Projects\C#\PackagingMSI>
D:\Projects\C#\PackagingMSI>msbuild Build.proj
Microsoft (R) Build Engine версии 12.0.31101.0
[Microsoft .NET Framework версии 4.0.30319.42000]
(C) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

Сборка начата 18.10.2015 16:19:27.
Проект "D:\Projects\C#\PackagingMSI\Build.proj" в узле 1 (целевые объекты по умолчанию).
Clean:
  Clean
Restore:
  Restore NuGet packages
  nuget.exe restore
  Все пакеты, перечисленные в packages.config, уже установлены.
Version:
  Versioning assemblies
Build:
  Build Release
Проект "D:\Projects\C#\PackagingMSI\Build.proj" (1) выполняет сборку "D:\Projects\C#\PackagingMSI\PackagingMSI\PackagingMSI.csproj" (2) в узле 1 (целевые объекты по умолчанию).
CoreResGen:
  Отсутствуют ресурсы, устаревшие по отношению к файлам своих источников. Пропуск формирования ресурсов.
GenerateTargetFrameworkMonikerAttribute:
  Целевой объект "GenerateTargetFrameworkMonikerAttribute" пропускается, так как все выходные файлы актуальны по отношению к входным.
CoreCompile:
  C:\Program Files (x86)\MSBuild\12.0\bin\Csc.exe /noconfig /nowarn:1701,1702 /nostdlib+ /platform:anycpu32bitpreferred /errorreport:prompt /warn:4 /define:TRACE /highentropyva+ /reference:"C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.5\Microsoft.CSharp.dll" /reference:"C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.5\mscorlib.dll" /reference:"C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.5\System.Core.dll" /reference:"C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.5\System.Data.DataSetExtensions.dll" /reference:"C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.5\System.Data.dll" /reference:"C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.5\System.Deployment.dll" /reference:"C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.5\System.dll" /reference:"C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.5\System.Drawing.dll" /reference:"C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.5\System.Windows.Forms.dll" /reference:"C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.5\System.Xml.dll" /reference:"C:\Program Files (x86)\Reference Assemblies\Microsoft\Framework\.NETFramework\v4.5\System.Xml.Linq.dll" /debug:pdbonly /filealign:512 /optimize+ /out:obj\Release\PackagingMSI.exe /subsystemversion:6.00 /resource:obj\Release\PackagingMSI.PackagingMSI.resources /resource:obj\Release\PackagingMSI.Properties.Resources.resources /target:winexe /utf8output MainForm.cs MainForm.Designer.cs Program.cs Properties\AssemblyInfo.cs Properties\Resources.Designer.cs Properties\Settings.Designer.cs "C:\temp\.NETFramework,Version=v4.5.AssemblyAttributes.cs"
CopyAppConfigFile:
  Целевой объект "_CopyAppConfigFile" пропускается, так как все выходные файлы актуальны по отношению к входным.
CopyFilesToOutputDirectory:
  Копирование файла из "obj\Release\PackagingMSI.exe" в "bin\Release\PackagingMSI.exe".
  PackagingMSI -> D:\Projects\C#\PackagingMSI\PackagingMSI\bin\Release\PackagingMSI.exe
  Копирование файла из "obj\Release\PackagingMSI.pdb" в "bin\Release\PackagingMSI.pdb".
Сборка проекта "D:\Projects\C#\PackagingMSI\PackagingMSI\PackagingMSI.csproj" завершена (целевые объекты по умолчанию).

Сборка проекта "D:\Projects\C#\PackagingMSI\Build.proj" завершена (целевые объекты по умолчанию).

Сборка успешно завершена.
Предупреждений: 0
Ошибок: 0

```

Target Pack will be started separately in command line **msbuild /target:Pack Build.proj**

```

Администратор: Developer Command Prompt for VS2013
D:\Projects\C#\PackagingMSI>msbuild /target:Pack Build.proj
Microsoft (R) Build Engine версии 12.0.31101.0
[Microsoft .NET Framework версии 4.0.30319.42000]
(C) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

Сборка начата 18.10.2015 16:25:35.
Проект "D:\Projects\C#\PackagingMSI\Build.proj" в узле 1 (целевые объекты Pack).
Pack:
  Pack application into MSI
  cscs.exe PackagingMSI.WixSharp/Script.cs PackagingMSI D:\Projects\C#\PackagingMSI
  C# Script execution engine. Version 3.9.16.0.
  Copyright (C) 2004-2014 oleg Shilo.

  Hello World!
Сборка проекта "D:\Projects\C#\PackagingMSI\Build.proj" завершена (целевые объекты Pack).

Сборка успешно завершена.
Предупреждений: 0
Ошибок: 0

```



## 4. Write packaging script

```
using System;
using System.Diagnostics;
//css_ref %WIXSHARP_DIR%\wixsharp.dll;
using WixSharp;

public class Script
{
    static public void Main(string[] args)
    {
        string projectName = args[0];
        string projectNameExe = projectName + ".exe";
        string projectFolder = args[1];
        string binaryFolder = projectFolder + @"\" + projectName + @"\bin\Release\";
        string assemblyPath = binaryFolder + projectNameExe;
        FileVersionInfo assemblyInfo = FileVersionInfo.GetVersionInfo(assemblyPath);
        Version version = new Version(assemblyInfo.FileVersion);

        Console.WriteLine("Project name: " + projectName);
        Console.WriteLine("Project folder: " + projectFolder);
        Console.WriteLine("Binary folder: " + binaryFolder);
        Console.WriteLine("Assembly path: " + assemblyPath);
        Console.WriteLine("Version: " + version.ToString());
        Console.WriteLine("Manufacturer: " + assemblyInfo.CompanyName);

        Project project =
            new Project(projectName + "_" + version.ToString(),
                new Dir(new Id("INSTALL_DIR"), @"%ProgramFiles%" + projectName,
                    new Files(binaryFolder + "*.exe"),
                    new Files(binaryFolder + "*.exe.config"),
                    new Files(binaryFolder + "*.dll"),

                    new Dir(@"%ProgramMenu%" + projectName,
                        new ExeFileShortcut(projectName, "[INSTALL_DIR]" + projectNameExe,
                            new ExeFileShortcut("Uninstall " + projectName, "[System64Folder]ms
                    ),
                    new Dir(@"%Startup%",
                        new ExeFileShortcut(projectName, "[INSTALL_DIR]" + projectNameExe,
                    )
                )
            );

        project.Version = version;
        project.GUID = new Guid("54f5a0b8-6f60-4a70-a095-43e2b93bc0fe");

        //project.SetNetFxPrerequisite("NETFRAMEWORK45 >='#378389'", "Please install .Net 4
        //project.ControlPanelInfo.ProductIcon = projectFolder + @"\" + projectName + @"\Re
        project.ControlPanelInfo.NoModify = true;
        project.ControlPanelInfo.Manufacturer = assemblyInfo.CompanyName;

        project.UI = WUI.WixUI_Common;
        var customUI = new CommomDialogsUI();
        var prevDialog = Dialogs.WelcomeDlg;
        var nextDialog = Dialogs.VerifyReadyDlg;
        customUI.UISequence.RemoveAll(x => (x.Dialog == prevDialog && x.Control == Buttons.
        customUI.On(prevDialog, Buttons.Next, new ShowDialog(nextDialog));
        customUI.On(nextDialog, Buttons.Back, new ShowDialog(prevDialog));
```



```
        project.CustomUI = customUI;  
  
        Compiler.BuildMsi(project);  
    }  
}
```

Do the following finale modifications:

Use same GUID in Script.cs as generated in AssemblyInfo.cs. (Or create a new GUID, but not use this one)

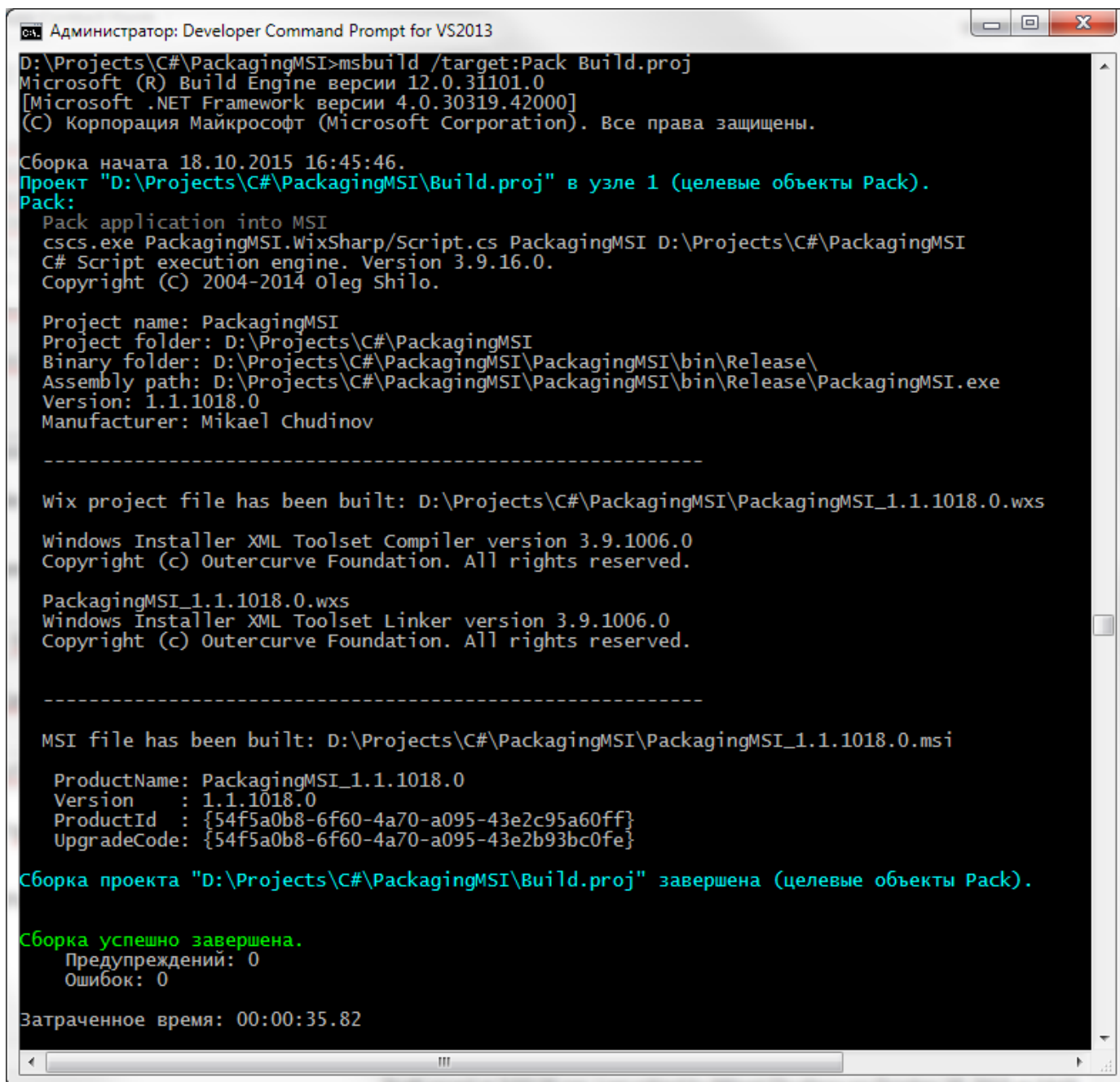
```
project.GUID = new Guid("54f5a0b8-6f60-4a70-a095-43e2b93bc0fe");
```

Then add AssemblyCompany info to assembly in **AssemblyInfo.cs**:

```
[assembly: AssemblyCompany("MyCompanyName")]
```

This is a requirement of MSI package.

Test msbuild /target:Pack Build.proj



```
Администратор: Developer Command Prompt for VS2013
D:\Projects\C#\PackagingMSI>msbuild /target:Pack Build.proj
Microsoft (R) Build Engine версии 12.0.31101.0
[Microsoft .NET Framework версии 4.0.30319.42000]
(C) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

Сборка начата 18.10.2015 16:45:46.
Проект "D:\Projects\C#\PackagingMSI\Build.proj" в узле 1 (целевые объекты Pack).
Pack:
  Pack application into MSI
  cscs.exe PackagingMSI.WixSharp/Script.cs PackagingMSI D:\Projects\C#\PackagingMSI
  C# Script execution engine. Version 3.9.16.0.
  Copyright (C) 2004-2014 Oleg Shilo.

  Project name: PackagingMSI
  Project folder: D:\Projects\C#\PackagingMSI
  Binary folder: D:\Projects\C#\PackagingMSI\PackagingMSI\bin\Release\
  Assembly path: D:\Projects\C#\PackagingMSI\PackagingMSI\bin\Release\PackagingMSI.exe
  Version: 1.1.1018.0
  Manufacturer: Mikael Chudinov

  -----

  Wix project file has been built: D:\Projects\C#\PackagingMSI\PackagingMSI_1.1.1018.0.wxs

  Windows Installer XML Toolset Compiler version 3.9.1006.0
  Copyright (c) Outercurve Foundation. All rights reserved.

  PackagingMSI_1.1.1018.0.wxs
  Windows Installer XML Toolset Linker version 3.9.1006.0
  Copyright (c) Outercurve Foundation. All rights reserved.

  -----

  MSI file has been built: D:\Projects\C#\PackagingMSI\PackagingMSI_1.1.1018.0.msi

  ProductName: PackagingMSI_1.1.1018.0
  Version : 1.1.1018.0
  ProductId : {54f5a0b8-6f60-4a70-a095-43e2c95a60ff}
  UpgradeCode: {54f5a0b8-6f60-4a70-a095-43e2b93bc0fe}

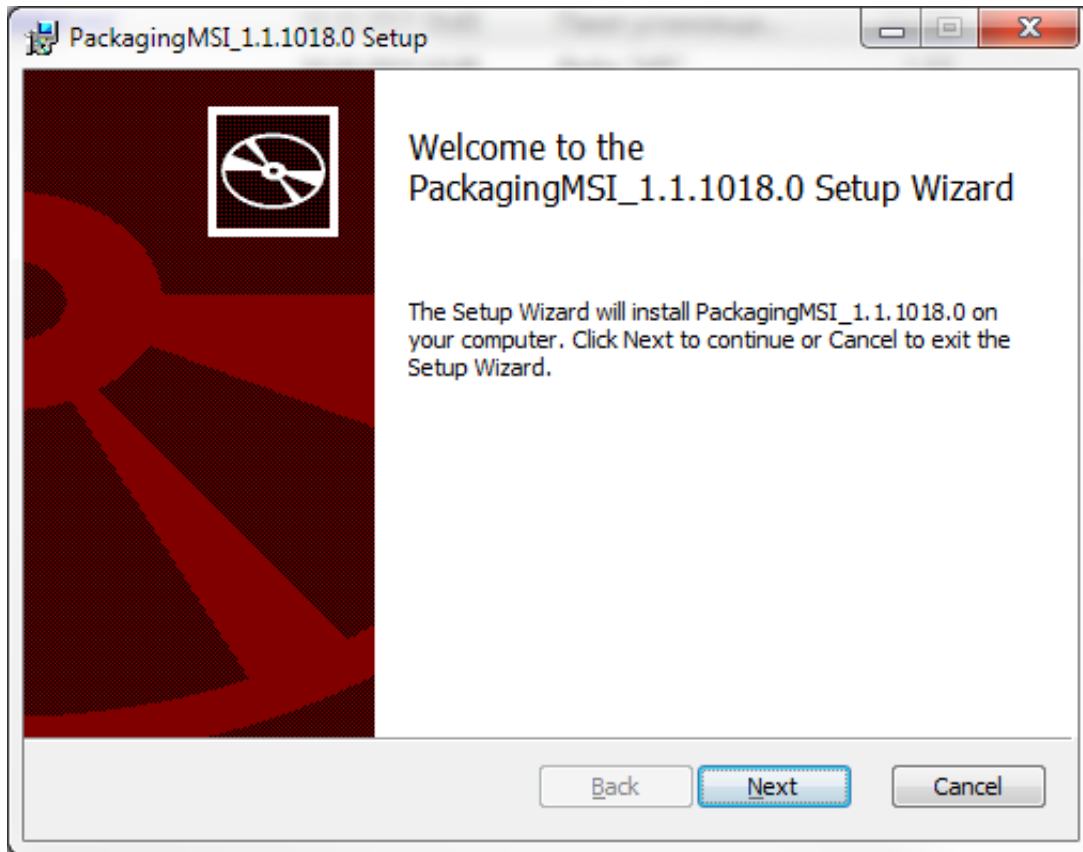
Сборка проекта "D:\Projects\C#\PackagingMSI\Build.proj" завершена (целевые объекты Pack).

Сборка успешно завершена.
  Предупреждений: 0
  Ошибок: 0

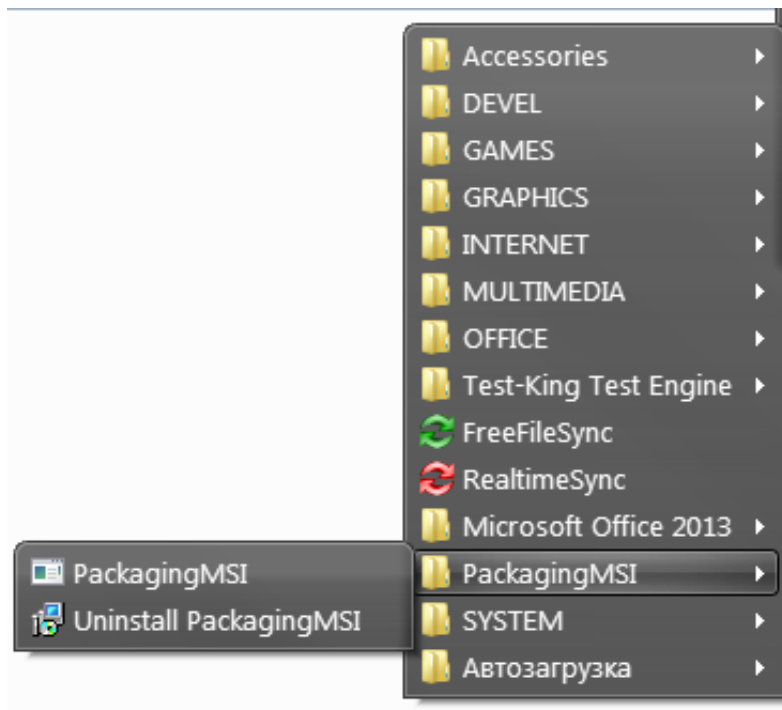
Затраченное время: 00:00:35.82
```

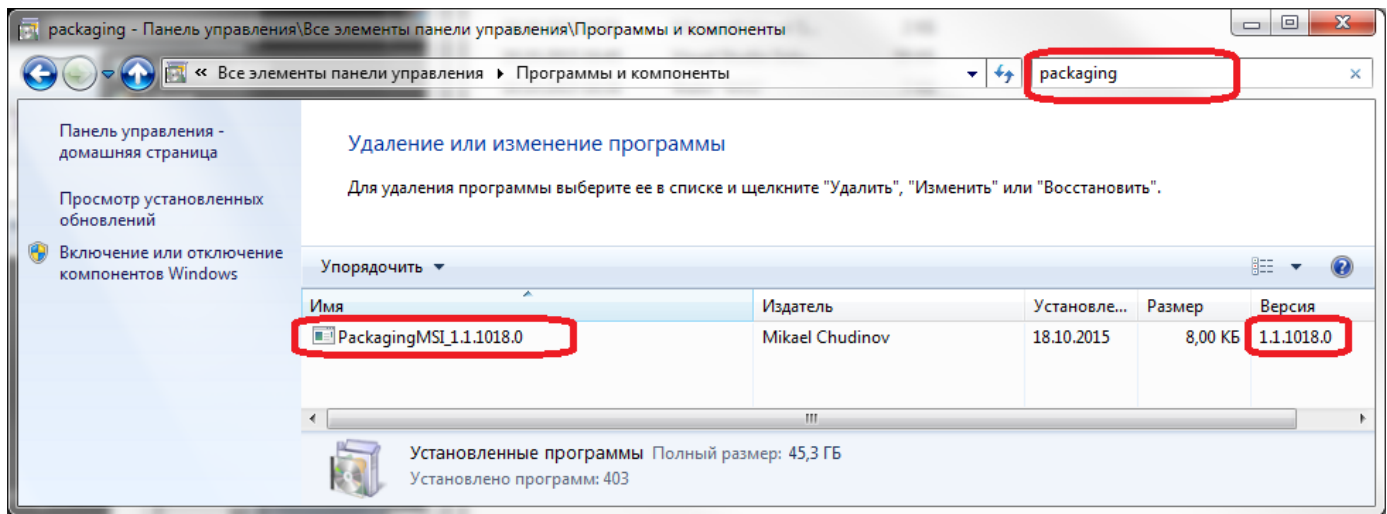
Script will create a new MSI package named <project\_name>.<version>.<daymonth>.msi in the solution folder.

Install this newly created package



Then program will be available in Programs menu and in Control Panel





### Share this:



This entry was posted in continuous integration and tagged .NET, continuous integration, packaging, Windows on May 18, 2015 [<http://blog.chudinov.net/packaging-of-a-net-application-on-windows/>] .