Visualization and Modeling SDK Lab – Part 5 / 6

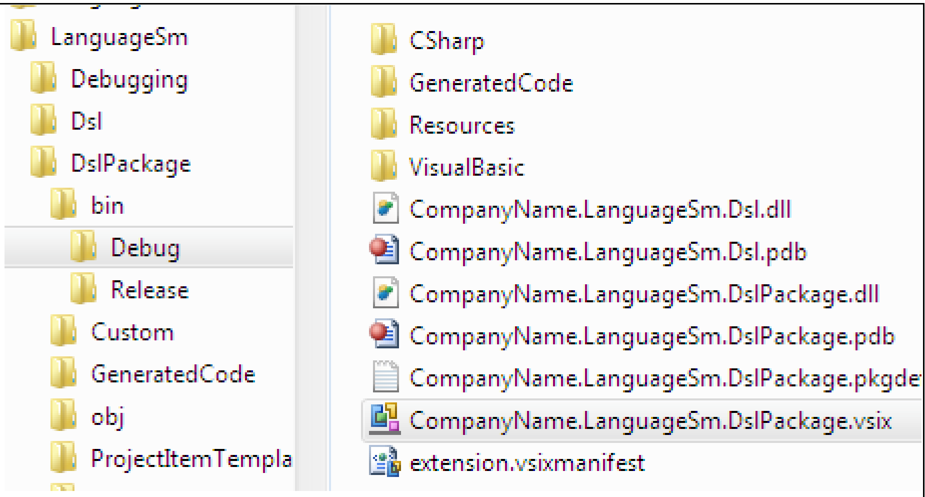
In the previous parts we showed you how to create a DSL by using the DSL Tools wizard (part 1), then we modified it to create our own metamodel and graphical syntax (part 2) for a final-state automaton. We then improved it (part 3), first by personalizing the UI, then by adding validation rules and validation of the model. Then (part 4), we generated the code that corresponds to our model, and created a custom tool.

We will finish this Lab by installing the DSL on any computer that has a copy of Visual Studio.

Deploying the VSIX

There are several ways in which you can improve your DSL, but when you are satisfied with it, you no doubt want to be able to offer it to your colleagues or customers. The DSLPackage project encapsulates the DSL in a Visual Studio Integration Extension (VSIX).

1. Build your DSL solution.
2. Find DslPackage\bin\**.vsix
   - In Visual Studio, right-click the tab at the top of any file window, and then click **Open Containing Folder**.
3. Copy the .vsix file to another computer on which Visual Studio is installed. (Not an Express edition.)
   - or –
   Leave it in place to install on your computer, so that it is available to the main instance of Visual Studio.
4. In Windows Explorer, double-click the file. Close and re-open Visual Studio.
5. You can now create and edit model files from the LanguageSm template, in a main instance of Visual Studio.
6. To uninstall the DSL, on the **Tools** menu, click **Extension Manager**.


LanguageSm
Debugging
Dsl
DslPackage
bin
Debug
Release
Custom
GeneratedCode
obj
ProjectItemTempla

CSharp
GeneratedCode
Resources
VisualBasic
CompanyName.LanguageSm.Dsl.dll
CompanyName.LanguageSm.Dsl.pdb
CompanyName.LanguageSm.DslPackage.dll
CompanyName.LanguageSm.DslPackage.pdb
CompanyName.LanguageSm.DslPackage.pkgde
CompanyName.LanguageSm.DslPackage.vsix
extension.vsixmanifest

You can add other components to the VSIX. For more information see Visual Studio Integration Extension (VSIX).

Deploying the VSIX in an MSI

There are several nice things we could do for our users:

- Provide an icon to identify our type of DSL file in Windows Explorer and Solution Explorer.
  (This is not the same as the VSIX icon, which appears in Extension Manager.)
- Associate our type of DSL file with Visual Studio, so that users can double-click a file in Windows, and have the file open in Visual Studio.
- Install an XSD so that the XML syntax of the DSL files is recognised by Visual Studio.

The scope of VSIX is restricted to extending Visual Studio, so that these features cannot be achieved with a VSIX. However, we can create an MSI (setup file) that can do these things, and can have the VSIX embedded within it.

Allow the VSIX to be deployed by MSI

1. Open DslPackage\source.extension.tt
2. Set InstalledByMsi true by inserting the line shown after Description.

```
<Vsix xmlns:xsi="...">
  <Identifier Id="<#= this.Dsl.PackageGuid #>">
    <Name><#= this.Dsl.DisplayName #></Name>
    <Author><#= this.Dsl.CompanyName #></Author>
    <Version><#= this.Dsl.MajorVersion #>.<#= this.Dsl.MinorVersion #>.<#=
this.Dsl.Build #>.<#= this.Dsl.Revision #></Version>
    <Description><#= this.Dsl.Description #></Description>
    <InstalledByMsi>true</InstalledByMsi>
```

Create a Setup solution

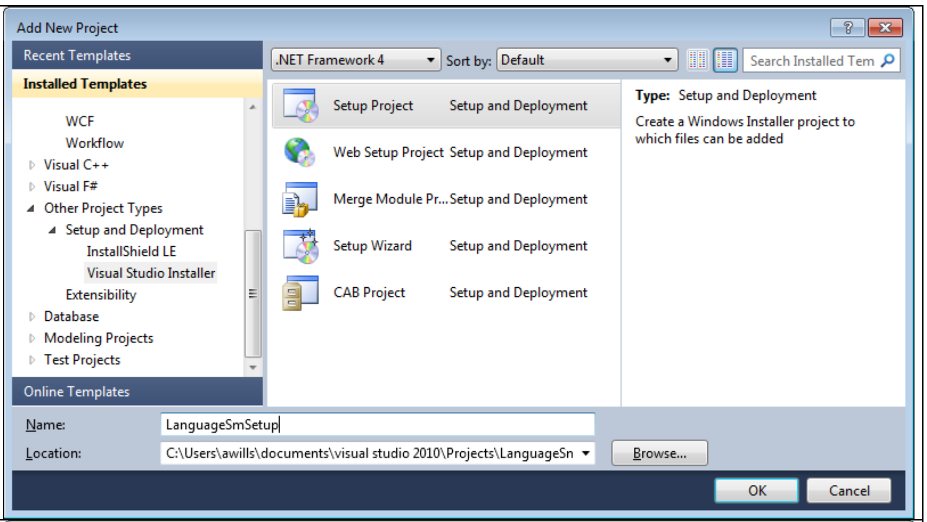
MSIs are created using Setup projects.

3. Add to the solution that contains

your DSL a new **Setup** project type in the **Other Project Types** -> **Setup and Deployment >** **Visual Studio Installer** category.

4. Name it **LanguageSmSetup**.

5. Click **OK**.



Add New Project

Recent Templates

Installed Templates

WCF
Workflow
Visual C++
Visual F#
Other Project Types
Setup and Deployment
InstallShield LE
Visual Studio Installer
Extensibility
Database
Modeling Projects
Test Projects

Online Templates

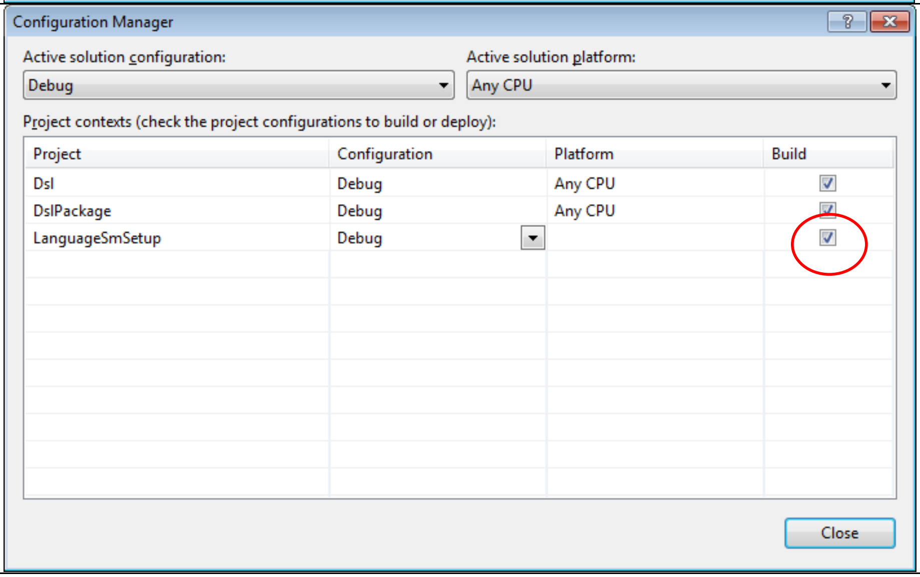
.NET Framework 4 | Sort by: Default | Search Installed Tem

Setup Project	Setup and Deployment
Web Setup Project	Setup and Deployment
Merge Module Pr...	Setup and Deployment
Setup Wizard	Setup and Deployment
CAB Project	Setup and Deployment

Type: Setup and Deployment
Create a Windows Installer project to which files can be added

Name: LanguageSmSetup
Location: C:\Users\awills\documents\visual studio 2010\Projects\LanguageSn
Browse...
OK | Cancel

6. Make sure that the new project is added to the build configuration. On the Build menu, click Configuration Manager.
7. Make sure there is a check mark alongside the new project.



Configuration Manager

Active solution configuration: Debug
Active solution platform: Any CPU

Project contexts (check the project configurations to build or deploy):

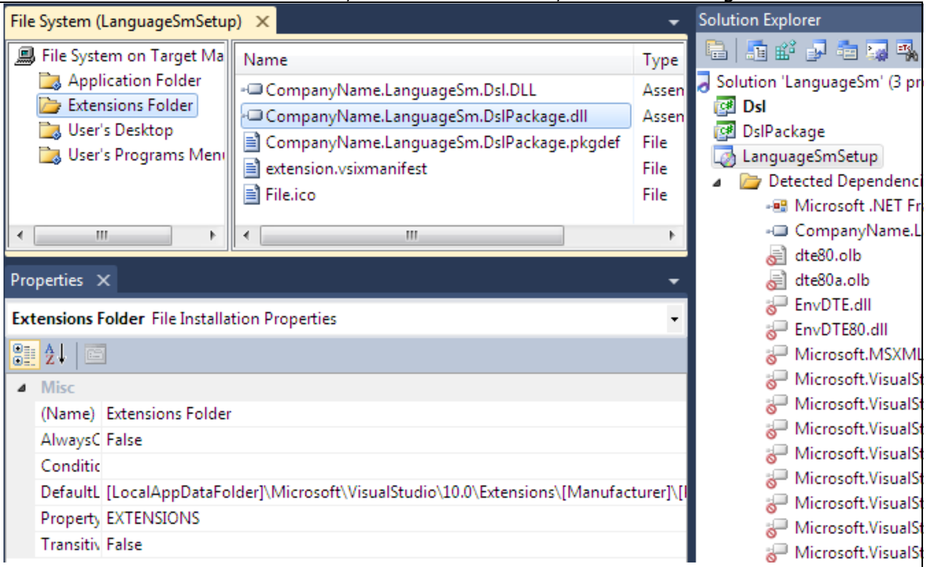
Project	Configuration	Platform	Build
Dsl	Debug	Any CPU	✓
DslPackage	Debug	Any CPU	✓
LanguageSmSetup	Debug		✓

Close

Target the DSL content at the Extensions Folder

The MSI mechanism will install the DSL VSIX content, instead of the VSIX installer. To do this, we have the MSI install the files in the Visual Studio Extensions folder. The files to install are the DLLs, the VSIX manifest, and the Package Definition.

8. In Solution Explorer, right-click LanguageSmSetup, point to **View** and click **File System**.
9. Right-click **File System on Target Machine**, **Add Special Folder** > **Custom Folder**.
10. Set the **Name** of the custom folder to Extensions Folder.
11. Set the **Default Location** of Extensions Folder to: [LocalAppDataFolder]\Microsoft\VisualStudio\10.0\Extensions\[Manufacturer]\[ProductName]\1.0.0.0
12. On Extensions Folder, use **Add** > **File** to add these files from DslPackage\binDebug:
    YourDsl.DslPackage.dll
    YourDsl.DslPackage.pkgdef
    extension.vsixmanifest
    DslPackage\Resources\File.ico
    Dependent files will also be added automatically.
13. Remove from the MSI dependent DLLs that will already exist in the target computers. This reduces the size of the MSI.
    In Solution Explorer, under LanguageSmSetup\Detected, select *all except* the following files



File System (LanguageSmSetup)

File System on Target Ma
Application Folder
Extensions Folder
User's Desktop
User's Programs Menu

Name	Type
CompanyName.LanguageSm.Dsl.DLL	Assen
CompanyName.LanguageSm.DslPackage.dll	Assen
CompanyName.LanguageSm.DslPackage.pkgdef	File
extension.vsixmanifest	File
File.ico	File

Properties

Extensions Folder File Installation Properties

Misc

(Name)	Extensions Folder
AlwaysC	False
Conditic	
DefaultL	[LocalAppDataFolder]\Microsoft\VisualStudio\10.0\Extensions\[Manufacturer]\[
Property	EXTENSIONS
Transitiv	False

Solution Explorer

Solution 'LanguageSm' (3 pr
Dsl
DslPackage
LanguageSmSetup
Detected Dependenc
Microsoft .NET Fr
CompanyName.L
dte80.olb
dte80a.olb
EnvDTE.dll
EnvDTE80.dll
Microsoft.MSXML
Microsoft.VisualS
Microsoft.VisualS
Microsoft.VisualS
Microsoft.VisualS
Microsoft.VisualS
Microsoft.VisualS
Microsoft.VisualS
Microsoft.VisualS

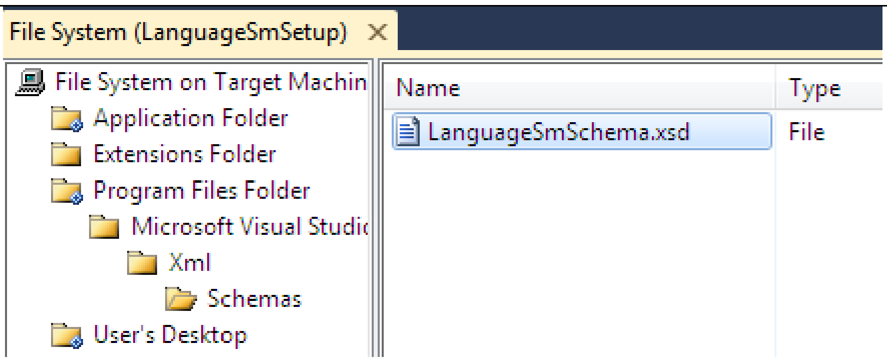
and right click **Exclude**:
Microsoft .Net Framework
YourDsl.Dsl.dll
Microsoft.VisualStudio.Modeling*,
unless you will always install on a computer that has VMSDK.

14. Edit File.ico to be the icon you want to represent your files in Explorer.

Target the XSD at the Visual Studio XML Folder

By installing the XSD of the DSL in the Visual Studio XML folder, we allow XML readers to recognize the DSL and avoid a warning message when models are opened.

15. Still in the File System window, right click **File System** > **Add Special Folder** > **Program Files Folder**.
16. Right click the **Program Files Folder** and repeatedly use **Add**>**Folder** to create a path: Microsoft Visual Studio 10.0\Xml\Schemas
17. To the Schemas folder, **Add**> **File**: DslPackage\bin\Debug\*Schema.xsd


File System (LanguageSmSetup)

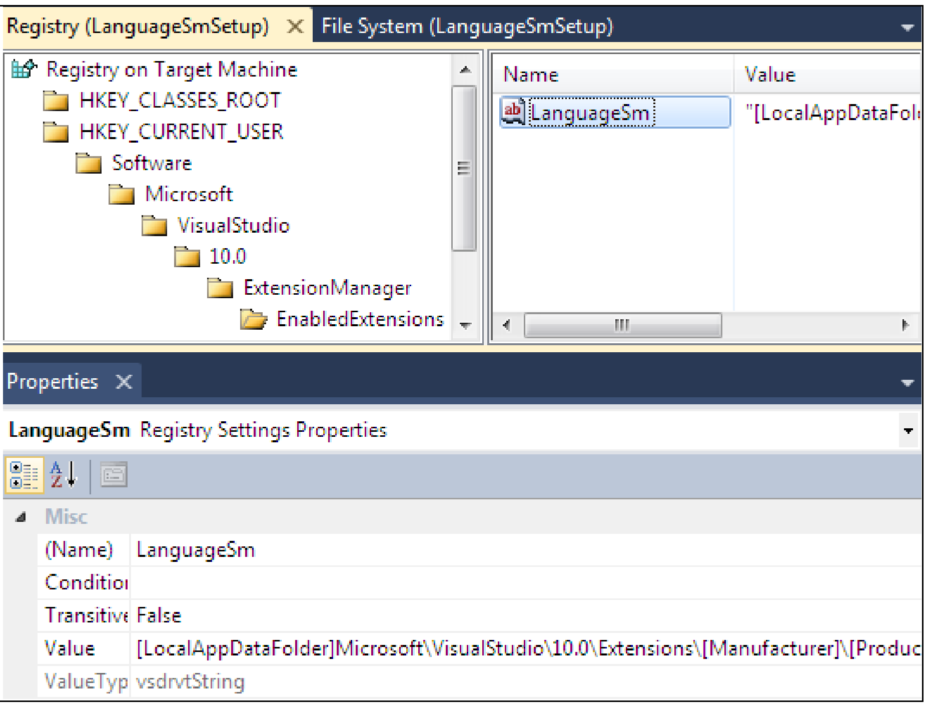
File System on Target Machin
- Application Folder
- Extensions Folder
- Program Files Folder
  - Microsoft Visual Studi
    - Xml
      - Schemas
- User's Desktop

Name	Type
LanguageSmSchema.xsd	File

Ensure the VSIX is Enabled on Installation

Without this step, the DSL is visible in the Exensions Manager, but cannot be enabled because we set the InstalledByMSI property.

18. Open the Registry view: in solution explorer, right-click LanguageSmSetup and click **View** > **Registry**.
19. Under HKEY_CURRENT_USER< repeatedly use **New**>**Key** to add: Software\Microsoft\VisualStudio\10.0\ExtensionManager\EnabledExtensions
20. Use **New** > **String Value** to create a string. Set the **Name** to the name of your DSL, LanguageSm. (This is the name at the root in DSL Explorer when you open the DSL Definition.)
Set the **Value** to: [LocalAppDataFolder]Microsoft\VisualStudio\10.0\Extensions\[Manufacturer]\[ProductName]\1.0.0.0


Registry (LanguageSmSetup) File System (LanguageSmSetup)

Registry on Target Machine
- HKEY_CLASSES_ROOT
- HKEY_CURRENT_USER
  - Software
    - Microsoft
      - VisualStudio
        - 10.0
          - ExtensionManager
            - EnabledExtensions

Name	Value
LanguageSm	"[LocalAppDataFol

Properties

LanguageSm Registry Settings Properties

Misc

(Name)	LanguageSm
Conditioı	
Transitive	False
Value	[LocalAppDataFolder]Microsoft\VisualStudio\10.0\Extensions\[Manufacturer]\[Produc
ValueTyp	vsdrvtString

Register the File Type

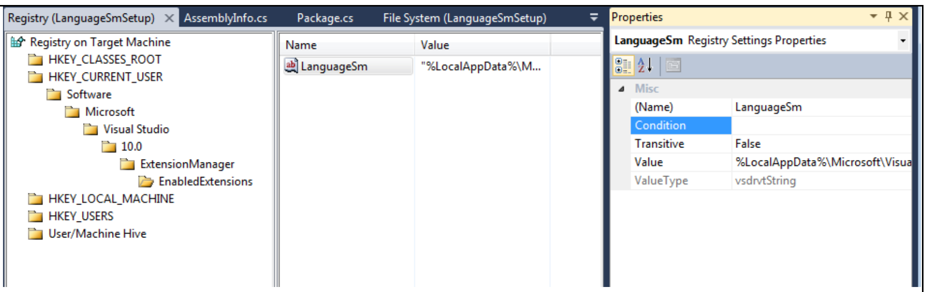
This step sets the icon and description that you see in Windows Explorer, and associates your DSL with Visual Studio so that you can open model files directly from Explorer.

2 Right-click LanguageSmSetup, point to **View**, then click **Registry**.

3 Under HKey_CURRENT_USER, repeatedly use **New>Key** to add Software\Microsoft\VisualStudio\10.0\ExtensionManager\EnabledExtensions
Remove any existing key.

4 Use New>String Value to append a string named LanguageSm with value [LocalAppDataFolder]Microsoft\VisualStudio\10.0\Extensions\[Manufacturer]\[ProductName]\1.0.0.0

5 Under HKEY_CLASSES_ROOT, add keys and subkeys according to the table at the right.
To add the (Default) strings, use **New>String Value** and then delete the value's Name.



Name	Value
LanguageSm	"%LocalAppData%\M...

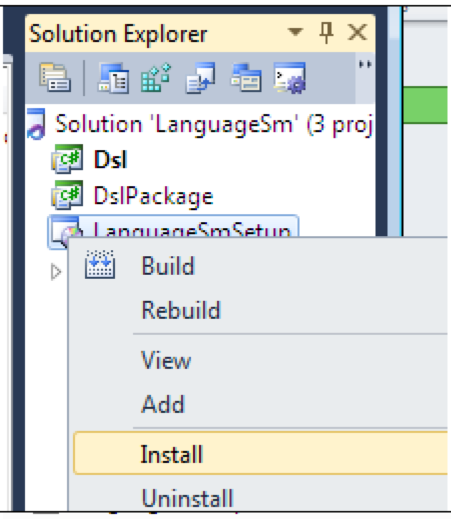
(Name)	LanguageSm
Condition	
Transitive	False
Value	%LocalAppData%\Microsoft\Visua
ValueType	vsdrvtString

Key	Default string
\.sm *(note the dot)*	LanguageSm
\LanguageSm	State Machine File
\LanguageSm\DefaultIcon	[LocalAppDataFolder]Microsoft\VisualStudio\10.0\Extensions\[Manufacturer]\[ProductName]\1.0.0.0\File.ico,0
\LanguageSm\Shell\Open\Command	"[ProgramFilesFolder]Microsoft Visual Studio 10.0\Common7\IDE\devenv.exe" %1

Let's try the setup!

6 Rebuild the solution.

7 In LanguageSmSetup\Debug, double-click the MSI file.
(For testing, you can alternatively right-click the LanguageSmSetup project in Solution Explorer and click **Install**.)

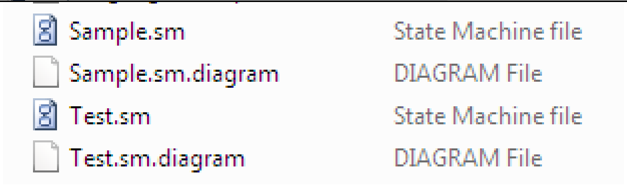


Tests
Verify the icon and description:
In Windows Explorer, open a folder that contains an **.sm** file – for example, the one in the Debugging folder. Verify that:

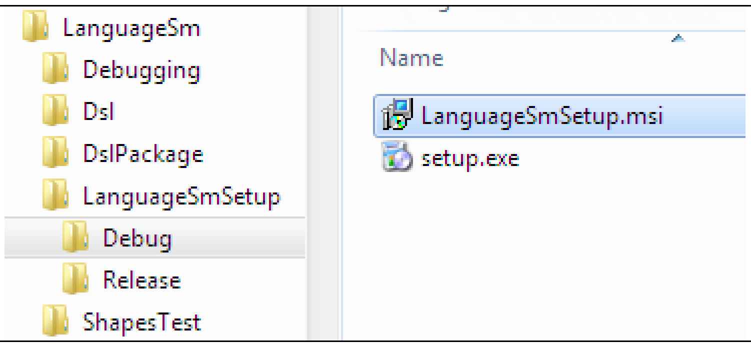
- The file is displayed with the icon you specified in File.ico.
  You might have to restart Windows Explorer to see this. Use Task Manager to stop explorer, then on the **File** menu click **New Task** and type "explorer".
- When you select **Details** in the **View** menu, the description of the file in the type column is the description you specified, State Machine File.
- Close all instances of Visual Studio. Double-click the .sm file. It should open in Visual Studio, as a diagram.

Verify that the XSD is correctly installed:

- When the .sm file opens in Visual Studio, there should be no errors or warnings.
- In Visual Studio, re-open the file



Sample.sm	State Machine file
Sample.sm.diagram	DIAGRAM File
Test.sm	State Machine file
Test.sm.diagram	DIAGRAM File

with the XML text editor. Attempt to insert new nodes. Verify that IntelliSense prompts you with the appropriate node names. NOTE: After you have opened a file with the XML editor, you must again use **Open With** to open the file in the default editor, as a diagram. Test the InstalledByMSI directive: • On the Tools menu, click Extension Manager. Verify that you can see the extension, but that you cannot disable or uninstall it from here.	
To uninstall, again double-click the generated MSI, or use the **Uninstall** command on the Setup project, or use Control Panel\Programs and Features.	
To test deployment on another computer, copy the MSI file from the Setup\Debug folder to the new computer. Double-click to install.	 LanguageSm Debugging Dsl DslPackage LanguageSmSetup Debug Release ShapesTest Name LanguageSmSetup.msi setup.exe