Release Notes VSPAT 1.3.20.0

This major release of VSPAT is a compatibility release for Visual Studio 2012. This release introduces two new versions of VSPAT: **VSPAT for Visual Studio 2010**, and **VSPAT for Visual Studio 2012**.

These two releases are deployed on the Visual Studio Gallery as two separate downloads, we strongly recommend moving to this latest version, as several critical changes have been made to transition to the future of VSPAT and pattern toolkits.

**IMPORTANT NOTICE:** This version of VSPAT, and any toolkits built with a previous version of VSPAT, are not forwards compatible with this version of VSPAT.

To upgrade to this release of VSPAT, existing Pattern Toolkit projects are required to be manually migrated for running in Visual Studio 2010, or running Visual Studio 2012.

Existing pattern toolkits that are not migrated may not work correctly with this version of VSPAT, or with other toolkits built with this version of VSPAT.

This document includes the sections ‘[Who Needs to Migrate?](#_Who_Needs_to)’ and ‘[Migration FAQ](#_Migration_FAQ)’ that should address any questions you may have about compatibility and migration of existing toolkits.

# What’s New?

* This version of VSPAT supports both Visual Studio 2012, and Visual Studio 2010. There will be two separate install packages, one for each version of Visual Studio.
* The toolsets ownership, identity and roadmap have been changed.
* Very minor bug fixes have been addressed.
* Pattern Toolkits that were built with a previous version of VSPAT (version 1.2.19.0 or earlier) will ultimately need to be migrated and rebuilt with this version of VSPAT in order to work with future versions of VSPAT.

This version of VSPAT now embeds the ‘Feature Extension Runtime’ extension which was previously installed as a separate extension into Visual Studio 2010.

The ‘Feature Extension Runtime’ extension can be downloaded and installed in the ‘Feature Builder Power Tool’, or by any ‘Feature Extensions’ built with the power tool, and was also installed by previous version of VSPAT in Visual Studio 2010. The ‘Feature Extension Runtime’ is no longer supported as a standalone extension.

**IMPORTANT**: We strongly recommended that you uninstall or disable the ‘Feature Builder Power Tool’ and ‘Feature Extension Runtime’ when upgrading to this version of VSPAT, or when installing any toolkit built with this version of VSPAT.

# Who Needs to Migrate?

Please use the following as guidance for determining whether to update existing pattern toolkits built with previous versions of VSPAT (prior to version 1.3.20.0).

**Note**: New pattern toolkits created with this version of VSPAT will NOT require migration, and will target the version of Visual Studio with which they were built.

You will need to migrate an existing pattern toolkit in either of the following scenarios:

1. As an author of a toolkit (built against a previous version VSPAT), you wish your toolkit to work with the new 1.3.20.0 version of VSPAT, in either Visual Studio 2010, or in Visual Studio 2012.
   * To upgrade your toolkit you will be required to:
     1. Uninstall the current version of VSPAT (i.e. the ‘Pattern Toolkit Builder’ extension, ‘Pattern Toolkit Library Support’ extension, and ‘Pattern Toolkit Runtime’ extension)
     2. Install the latest version of ‘Pattern Toolkit Builder’ extension.
     3. Migrate the pattern toolkit to the latest version of VSPAT (using the notes at the end of this document)
2. As a user of a toolkit (built against a previous version VSPAT), you wish your toolkit to work with the new 1.3.20.0 version of VSPAT, in either Visual Studio 2010, or in Visual Studio 2012.

**Important**: Toolkits that were built with previous versions of VSPAT will continue to install and work in Visual Studio 2010, and will not require immediate migration, provided the user of the toolkit does not install another toolkit that was built with a newer version of VSPAT into their Visual Studio.

In development projects where the development tools are strictly controlled this will likely not be an issue. However, for development environments that are not strictly controlled, where developers are downloading and updating their own extensions for Visual Studio, this will ultimately be an issue.

It is therefore strongly recommended that all toolkits built with previous versions of VSPAT are eventually migrated to avoid backwards compatibility issues.

* + To upgrade your toolkit you will be required to:
    1. Obtain a newer version of your pattern toolkit from the original author of the toolkit.
    2. Uninstall your existing pattern toolkit.
    3. Uninstall all VSPAT related extensions (i.e. the ‘Pattern Toolkit Runtime’ extension)
    4. Uninstall the ‘Feature Extension Runtime’ extension.
    5. Manually install the latest version of your pattern toolkit, provided by the author of the toolkit.

**Note**: Neither of these kinds of upgrades will happen automatically for an existing pattern toolkit.

## Supported Migration Scenarios

There are 2 main migration scenarios for existing toolkits that are documented and supported in this version. Other scenarios may exist, but the reader will need to determine the detailed steps to execute them.

1. Migrate toolkit to run in Visual Studio 2010
2. Migrate toolkit to run in Visual Studio 2012

**Note**: Previous versions of VSPAT only supported Visual Studio 2010, the new version of VSPAT is supported in both Visual Studio 2010 and Visual Studio 2012.

**Note**: New versions of VSPAT may support the building of toolkits that run in both Visual Studio 2010 and Visual Studio 2012.

### Migrate toolkit to run in Visual Studio 2010

In this scenario you wish to migrate an existing toolkit (built with a previous version of VSPAT in Visual Studio 2010), for use in Visual Studio 2010.

* In this scenario, your built toolkit will be installed into Visual Studio 2010.
* In this scenario, your built toolkit will never be installed into Visual Studio 2012.

### Migrate toolkit to run in Visual Studio 2012

In this scenario you wish to migrate an existing toolkit (built with a previous version of VSPAT in Visual Studio 2010), for use in Visual Studio 2012.

* In this scenario, your built toolkit will be installed into Visual Studio 2012.
* In this scenario, your built toolkit will never be installed into Visual Studio 2010.

# Migration FAQ

### Do existing toolkits that were built with a previous version of VSPAT require migration?

Eventually yes, it is strongly recommended, see the previous section on [Who Needs to Migrate?](#_Who_Needs_to) for more details about this question, and then follow the [Migration Notes](#_Migration_Notes) section.

### What happens to existing toolkits built with a previous version of VSPAT?

**For Visual Studio 2010**: Existing toolkits will continue to install and work in Visual Studio 2010 as before, provided a new pattern toolkit built with this version of VSPAT (or later version) is not installed into Visual Studio. See [Who Needs to Migrate?](#_Who_Needs_to) for more details about why migration is still strongly recommended.

**For Visual Studio 2012**: Does not apply, as no previous version of VSPAT, or pattern toolkits supported Visual Studio 2012 installation.

### Will VSPAT update from the Visual Studio Gallery?

**For Visual Studio 2010**: Yes, authors of existing toolkits (or those who have already installed VSPAT) can be notified of the new version of VSPAT from the Visual Studio Gallery, and they can choose to install the new version. At which point, they will need to migrate their existing pattern toolkit projects.

**Note**: Users of existing toolkits will not be notified of updates from the Visual Studio Gallery.

**For Visual Studio 2012**: No, there was no previous version of VSPAT for Visual Studio 2012 to update.

### Can I target my toolkit to run in Visual Studio 2012?

**For Visual Studio 2010**: Toolkits built with Visual Studio 2010 will not run properly in Visual Studio 2012 because these toolkits have dependencies on Visual Studio 2010 that are not present in Visual Studio 2012. If you want your toolkit to run in Visual Studio 2012, you must build (or migrate) your toolkit in Visual Studio 2012.

**For Visual Studio 2012**: All toolkits built in Visual Studio 2012 are targeted to run correctly in Visual Studio 2012. But these toolkits will not install or work correctly in Visual Studio 2010.

### How do I build my toolkit to target both VS2010 and VS2012?

This is currently not supported in this version of VSPAT. Multi-targeting toolkits is limited to general issues in multi-targeting any VSIX in Visual Studio. A toolkit is simply just another VSIX extension. However, a toolkit currently has build dependencies, and sub-dependencies that are not necessarily available in both versions of Visual Studio. Whilst multi-targeting a toolkit may be technically possible, it is not currently supported in this version of VSPAT. The VSPAT project is investigating future approaches and techniques that may help toolkit builders target their toolkit to either Visual Studio 2010 or Visual Studio 2012 using the same codebase of a toolkit.

### Technically, why do we need to migrate toolkits, what has changed in this version?

This new version of VSPAT has had to undergo a number of major changes in its binary compatibility, in its dependencies, and its ownership.

This version of VSPAT now addresses a new version of Visual Studio 2012, which brings changes in how toolkits (VSIX extensions) are built and deployed. Generally speaking, a Visual Studio eXtension (VSIX) that is built with Visual Studio 2010 is not compatible with Visual Studio 2012 without some changes to its dependencies and registration information. It is because of these compatibility issues that two versions of the VSPAT extensions will need to be deployed, targeted separately at Visual Studio 2010 and at Visual Studio 2012.

In addition, a major dependency of VSPAT to date has been the ‘Feature Extension Runtime’ extension, which is a sub-component of the ‘Feature Builder Power Tool’ extension. Support for the ‘Feature Builder Power Tool’ extension from Microsoft has ended, and there will be no release of the Power Tool for Visual Studio 2012. Therefore, this dependency has necessarily been packaged into the VSPAT extension. The version of the ‘Feature Extension Runtime’ contained within this version of VSPAT is not compatible with previous versions of the ‘Feature Builder Power Tool’ nor the ‘Feature Extension Runtime’ extension.

In addition, the ownership of the VSPAT project has recently been transferred to ‘The Outercurve Foundation’ where it benefits from being a supported open source project. This has necessarily had to change the registration and ownership and identification of the binary deliverables of VSPAT, and again these are not compatible with previous versions of VSPAT.

These changes have led to a new major version of VSPAT being released that is unfortunately not backwardly compatible with previous versions of VSPAT, and not compatible with toolkits built with a previous version of VSPAT. The long term goal of this version of VSPAT is to replace previous installations of VSPAT and upgrade any versions of toolkits built with previous versions of VSPAT. Once this version migration is complete, no such detailed migration will be required again.

As this version of VSPAT is not backwardly compatible with previous versions, any toolkits that were built with previous versions of VSPAT will no longer work correctly if installed alongside toolkits that are built with this version of VSPAT. This is why we strongly recommend migration of any existing toolkits.

### What happens to existing toolkits that already embed the ‘Pattern Toolkit Manager’ from 1.2.19.0?

**For Visual Studio 2010**: If a Visual Studio 2010 user already has a toolkit installed into Visual Studio, and then either installs VSPAT (‘Pattern Toolkit Builder’ extension) or installs a pattern toolkit built with the new version of VSPAT, then the older version of the ‘Pattern Toolkit Manger’ will be upgraded automatically for them. Their toolkit should continue to work as before.

**For Visual Studio 2012**: Does not apply, as no previous version of VSPAT, or pattern toolkits supported Visual Studio 2012 installation.

### What happens if the ‘Feature Extension Runtime’ extension (or ‘Feature Builder Power Tool’ extension is already installed when I install this version of VSPAT?

**For Visual Studio 2010**: When Visual Studio 2010 starts, or while using Visual Studio, the following error is reported multiple times:



You must either [Disable] or [Uninstall] the ‘Feature Extension Runtime’ extension (and any and all extensions related to the ‘Feature Builder Power Tool’) in ‘Extension Manager’.



**For Visual Studio 2012**: Does not apply, as no previous version of the ‘Feature Extension Runtime’ or Feature Builder Power Tool’ supported Visual Studio 2012 installation.

# Migration Notes

These manual actions need to be followed in order to migrate existing pattern toolkits to the current version of VSPAT (v.1.3.20.0), from any previous version (**v.1.2.19.0** or earlier)**.**

Since this version of VSPAT supports both Visual Studio 2010 and Visual Studio 2012. The migration steps are different depending on which version of Visual Studio you wish to target your toolkit for.

**Note**: In this version of VSPAT, you must migrate your existing toolkit project to the version of Visual Studio you wish to target for your toolkit.

Individual migration steps with be marked for the Visual Studio version they apply, where there are differences.

* **VS2010** – Migration steps specific only to Visual Studio 2010
* **VS2012** – Migration steps specific only to Visual Studio 2012

**Important**: Before starting the migration , it you are strongly recommend to backup your source code, or use source control, to manage any changes you make in migrating your toolkits projects.

### Pattern Toolkit Solutions:

#### Pre Work:

**VS2012:**

* If your toolkit solution contains one or more DSL projects (unusual for most toolkit projects), we recommend you use the DSL Tool Migration wizard to first convert your DSL projects. Found in: **%ProgramFiles(x86)%\Microsoft Visual Studio 11.0\VSSDK\VisualStudioIntegration\Tools\DSLTools\DslProjectsMigrationTool.exe**. Please consult the **MigrationGuide.mht** in the same folder.
* If your toolkit solution has projects which use the following targets files (unusual for most toolkit projects)
  + Change: $(MSBuildExtensionsPath)\Microsoft\VisualStudio\TextTemplating\v10.0\Microsoft.TextTemplating.targets
  + To: $(MSBuildExtensionsPath)\Microsoft\VisualStudio\v11.0\TextTemplating\Microsoft.TextTemplating.targets

#### Solution file (\*.sln)

* + Open in ‘XML View’:
  + Modify the following:

GlobalSection(ExtensibilityGlobals) = postSolution

Features = 9f6dc301-6f66-4d21-9f9c-b37412b162f6:Creating Pattern Toolkits:1.3.20.0

EndGlobalSection

#### Solution Builder File (\*.slnbldr)

* + Open in ‘XML View’:
  + Modify any <product> elements where DefinitionName=”PatternToolkit” to the following:

<product definitionName="PatternToolkit" extensionName="Pattern Toolkit Builder" version="1.3.20.0">

### Pattern Toolkit Project:

#### Toolkit Project File (csproj):

* + Unload project, and edit the XML:
  + **VS2010:** Add the following “PropertyGroup” as the first “PropertyGroup” in the file:

<PropertyGroup>

<MinimumVisualStudioVersion>10.0</MinimumVisualStudioVersion>

<VisualStudioVersion Condition="'$(VisualStudioVersion)' == ''">10.0</VisualStudioVersion>

<VSToolsPath Condition="'$(VSToolsPath)' == ''">$(MSBuildExtensionsPath32)\Microsoft\VisualStudio\v$(VisualStudioVersion)</VSToolsPath>

</PropertyGroup>

* + **VS2012:** Add the following “PropertyGroup” and “ as the first “PropertyGroup” in the file:

<PropertyGroup>

<MinimumVisualStudioVersion>11.0</MinimumVisualStudioVersion>

<VisualStudioVersion Condition="'$(VisualStudioVersion)' == ''">11.0</VisualStudioVersion>

<VSToolsPath Condition="'$(VSToolsPath)' == ''">$(MSBuildExtensionsPath32)\Microsoft\VisualStudio\v$(VisualStudioVersion)</VSToolsPath>

</PropertyGroup>

* + **VS2012:** Add the following “PropertyGroup” and “Import” immediately below the first “PropertyGroup” in the file:

<PropertyGroup>

<IncludeAssemblyInVSIXContainer>false</IncludeAssemblyInVSIXContainer>

<IncludeDebugSymbolsInVSIXContainer>false</IncludeDebugSymbolsInVSIXContainer>

<IncludeDebugSymbolsInLocalVSIXDeployment>false</IncludeDebugSymbolsInLocalVSIXDeployment>

<CopyBuildOutputToOutputDirectory>false</CopyBuildOutputToOutputDirectory>

<CopyOutputSymbolsToOutputDirectory>false</CopyOutputSymbolsToOutputDirectory>

</PropertyGroup>

<Import Project="$(MSBuildExtensionsPath)\$(MSBuildToolsVersion)\Microsoft.Common.props" Condition="Exists('$(MSBuildExtensionsPath)\$(MSBuildToolsVersion)\Microsoft.Common.props')" />

* + **VS2012:** Change all “Microsoft.VisualStudio.\*.dll” version 10.0 assembly references to version 11.0 assembly references.

<Reference Include="Microsoft.VisualStudio.Shell, Version=11.0.0.0, Culture=neutral, PublicKeyToken=b03f5f7f11d50a3a, processorArchitecture=MSIL" />

<Reference Include="Microsoft.VisualStudio.ExtensibilityHosting, Version=11.0.0.0, Culture=neutral, PublicKeyToken=b03f5f7f11d50a3a, processorArchitecture=MSIL" />

* + **VS2012:** Change all “Microsoft.VisualStudio.\*.10.0.dll” assembly references to “Microsoft.VisualStudio.\*.11.0.dll” assembly references.

<Reference Include="Microsoft.VisualStudio.Shell.Immutable.11.0, Version=11.0.0.0, Culture=neutral, PublicKeyToken=b03f5f7f11d50a3a, processorArchitecture=MSIL" />

* + **VS2012:** Change the value of the “TargetFrameworkVersion” property:

<TargetFrameworkVersion>v4.5</TargetFrameworkVersion>

* Change the “Import” to the “Microsoft.VsSDK.targets”:

<Import Project="$(VSToolsPath)\VSSDK\Microsoft.VsSDK.targets" />

Reload the project

#### Toolkit Automation Project File (csproj):

* + Unload project, and edit the XML:
  + **VS2012:** Change all “Microsoft.VisualStudio.\*.dll” version 10.0 assembly references to version 11.0 assembly references.

<Reference Include="Microsoft.VisualStudio.ExtensibilityHosting, Version=11.0.0.0, Culture=neutral, PublicKeyToken=b03f5f7f11d50a3a, processorArchitecture=MSIL" />

* + **VS2012:** Change all “Microsoft.VisualStudio.\*.10.0.dll” assembly references to “Microsoft.VisualStudio.\*.11.0.dll” assembly references.

<Reference Include="Microsoft.VisualStudio.Shell.Immutable.11.0, Version=11.0.0.0, Culture=neutral, PublicKeyToken=b03f5f7f11d50a3a, processorArchitecture=MSIL" />

<Reference Include="Microsoft.VisualStudio.Modeling.Sdk.11.0, Version=11.0.0.0, Culture=neutral, PublicKeyToken=b03f5f7f11d50a3a, processorArchitecture=MSIL" />

<Reference Include="Microsoft.VisualStudio.Modeling.Sdk.Diagrams.11.0, Version=11.0.0.0, Culture=neutral, PublicKeyToken=b03f5f7f11d50a3a, processorArchitecture=MSIL" />

* + **VS2012:** Change the value of the “TargetFrameworkVersion” property:

<TargetFrameworkVersion>v4.5</TargetFrameworkVersion>

Reload the project

#### Source.extension.tt

* + **VS2012:** Change the version of the SupportedProducts\VisualStudio\@Version attribute:

<SupportedProducts>

<VisualStudio Version="11.0">

<Edition>Ultimate</Edition>

<Edition>Premium</Edition>

<Edition>Pro</Edition>

</VisualStudio>

</SupportedProducts>

#### All Project Template and Item Templates Files (\*.vstemplate):

* + Add version and update the 'PublicKeyToken' strings from all <WizardExtension> elements for assemblies beginning with ‘Microsoft.VisualStudio.Patterning’. e.g.

<WizardExtension>  
    <Assembly>Microsoft.VisualStudio.Patterning.Library, Version=1.3.20.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35</Assembly>

#### All Project Template Project files (\*.csproj)

**VS2012**: Add the following “Import” to the top of the file:

<Import Project="$(MSBuildExtensionsPath)\$(MSBuildToolsVersion)\Microsoft.Common.props" Condition="Exists('$(MSBuildExtensionsPath)\$(MSBuildToolsVersion)\Microsoft.Common.props')" />