Solution Builder Guidance

First Time Users

These topics are aimed at the first time users of Solution Builder.

Understanding

This section helps you understand what Solution Builder is, what Pattern Toolkits are, why they are useful, and who should use them.

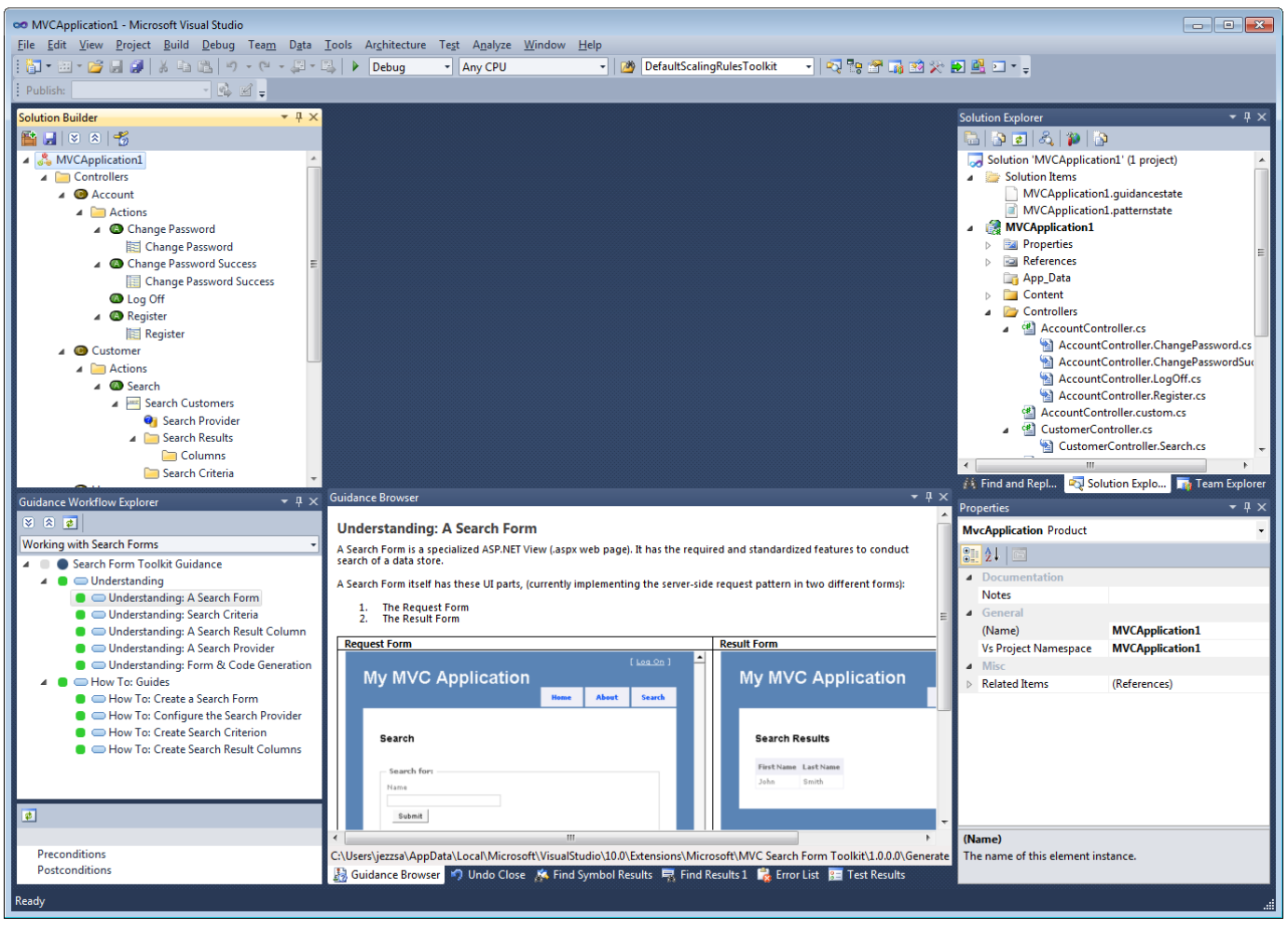
What is Solution Builder?

The 'Solution Builder' window, (CTRL + W, H) is the main window where you can view, manage and configure elements of your solution that are provided by 'Pattern Toolkit' extensions.

The elements displayed in this window are called 'Solution Elements', and they automate the development and deployment of software in the current solution.

In the 'Solution Builder' window works with some other windows to allow you to perform the following common activities:

- Create new solution elements.
- View, and Browse the existing solution elements for the current solution.
- Configure the properties of solution elements.
- Get help and guidance for using the solution elements for developing your solution.
- Compose new or existing solution elements together.
- Navigate to linked related artifacts elsewhere in the development environment.



1 An example Pattern Toolkit for developing ASP.NET MVC applications.

> **Note:** Think of 'Solution Elements', as being abstractions of data or artifacts in the current solution scope, whether they represent physical artifacts such as projects, files or solutions, or data from other services in the development environment. They represent a notion that has meaning in the context of the installed toolkit.

Where do the Solution Elements come from?

The different types of solution elements in this window are installed by Pattern Toolkits. Pattern Toolkits are extensions that are installed into Visual Studio either using the 'Extension Manager' in Visual Studio or by obtaining pattern toolkits from toolkit authors, such as from the Visual Studio Gallery. Pattern Toolkits are built with NuPattern (see http://nupattern.org for more details.

Using Pattern Toolkits

Pre-requisites for Using Pattern Toolkits

All that you need to start using a pattern toolkit is to install Visual Studio 2010/2012 Professional, Premium or Ultimate.

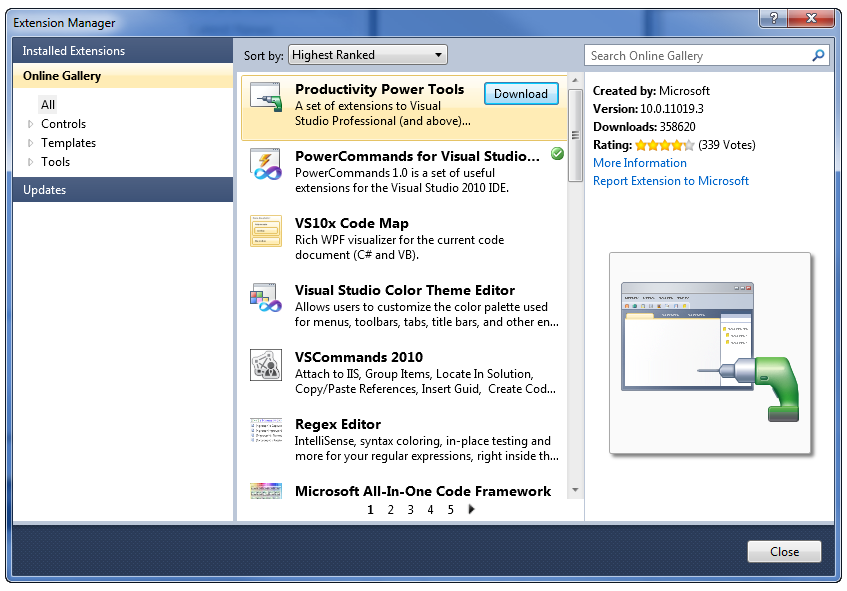
> **Note:** A pattern toolkit installs all its own pre-requisites and dependencies.

Installing a Pattern Toolkit

Pattern toolkits come in a *.vsix file. They can be downloaded directly and installed on a computer, or you can obtain them from within Visual Studio using the 'Extension Manager'.

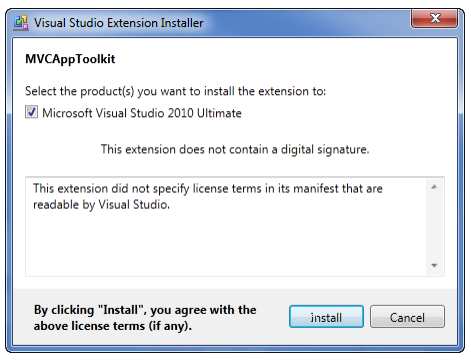
Install from Within Visual Studio

Pattern Toolkits that are published in the 'Visual Studio Gallery', can be downloaded and installed from within Visual Studio by using the 'Extension Manager'.



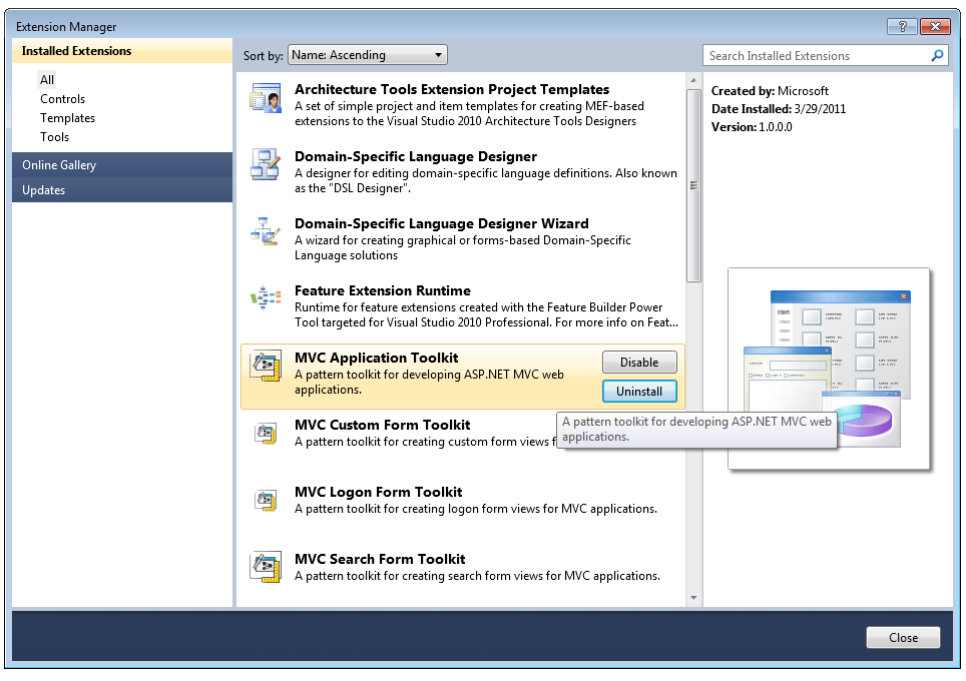
Install From File

To install a pattern toolkit that you have obtained as a *.vsix file from a toolkit author, simply open that file from disk, and click though the installer.



Browsing & Managing Installed Pattern Toolkits

You can browse, disable and uninstall the installed patterns toolkits in Visual Studio, using the Visual Studio 'Extension Manager'.



Note: When an extension is 'Disabled' or 'Uninstalled', a restart of Visual Studio is required for the change to take effect.

Using Pattern Toolkits

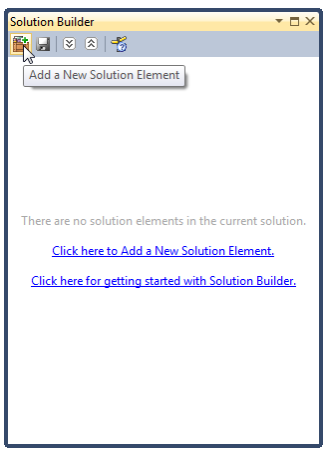
Once a pattern toolkit is installed, the contained pattern within it is available to be used from the 'Solution Builder' window.

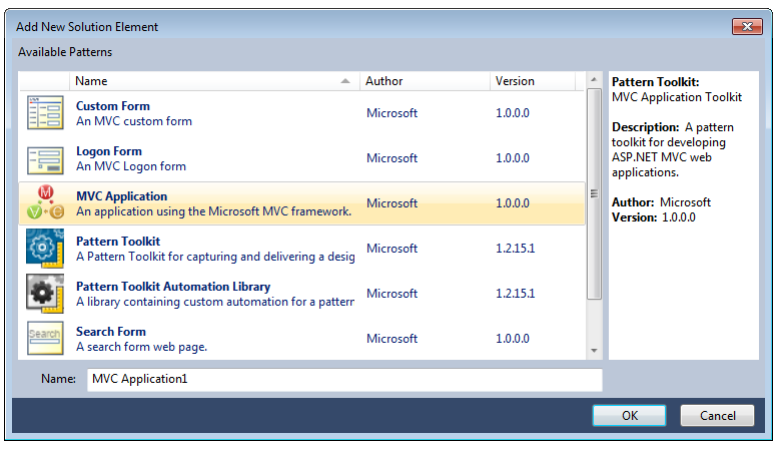
> **Note:** A user of a toolkit is really more focused on solving problems in their solution using whatever tools are available to them. Therefore once a toolkit is installed the user of the toolkits really just wants to use the elements within the toolkit to help them build their solution. For this reason, the terminology used from a user's perspective switches from patterns and toolkits to 'building solutions' with what are called 'Solution Elements'.

Creating new 'Solution Elements' is performed either directly from within the 'Solution Builder' tool window, or automatically from the more traditional method of creating new projects or items in Visual Studio using the 'Add New Project/Item' dialog, (Dependent on whether the particular Pattern Toolkit provides a project or item template for the pattern).

To create a new solution element in the **'Solution Builder'** tool window, click the 'Add New Solution Element' button.

This displays the 'Add New Solution Element' dialog, where you can choose to create from one of the patterns provided by the currently installed Pattern Toolkits.



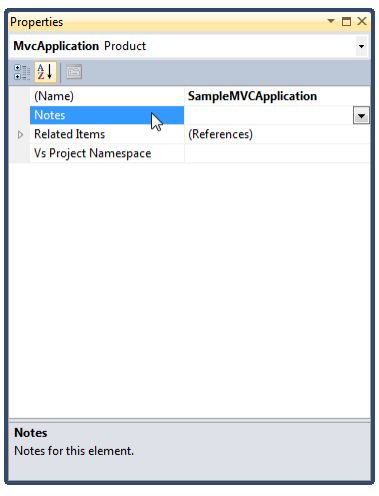


Viewing & Configuring Solution Elements

You configure and compose the 'Solution Elements' in your solution in the 'Solution Builder' window.

Depending on how the specific pattern toolkit is implemented for a specific solution element, various standard features may be available to you.

Editing General Properties of a Solution Element



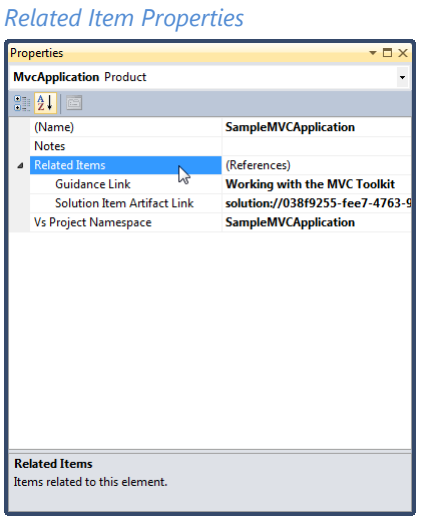
Selecting a solution element allows you to edit its properties in the 'Properties Window':

Editing Properties:

- Each property may have its own editor for its value, a dropdown of values or allow you to type a value.
- Some properties are read only.

All properties have a 'Description' pane below explaining their meaning.

Related Item Properties



All solution elements have a 'Related Items' property, that when populated, relate the solution element other items (e.g. solution artifacts, guidance workflows, model elements, source code elements etc.) within the Visual Studio environment or externally to Visual Studio. See Related Items for details.

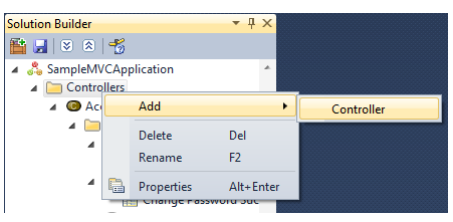
> **Note**: Depending on the type of related item, they can usually be modified (or fixed) to relate to other existing items, should the related item ever become un-related for whatever reason (i.e. deleted).

Notes Properties

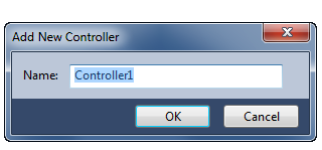
Every solution element has a 'Notes' property that allows you to leave design notes or annotations describing the current solution element.

Adding Child Solution Elements

If the solution element has child elements defined in the pattern, and allows more than one instance of that child element, then you can add more instances by right-clicking on the element and selecting the type of child to add from the 'Add' menu.

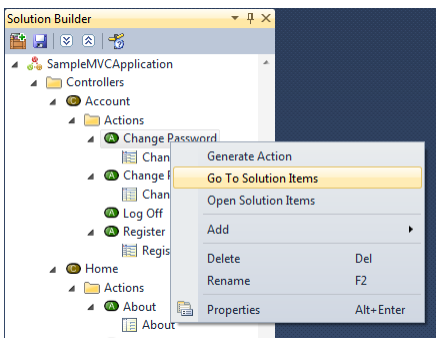


Once the menu is selected, the child element is named in the 'Add New' dialog:



Navigating to Related Solution Artifacts

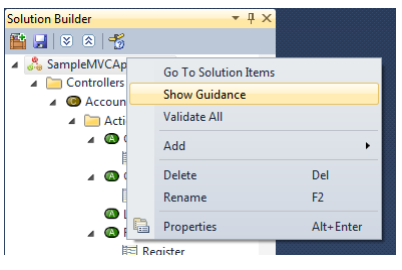
If the solution element has a 'Related Item' (one or more) that refer to solution artifact which are navigable in 'Solution Explorer' (called a 'Solution Item Artifact Link'), then you can navigate to them using the 'Go to Solution Item' menu.



If the solution element has a 'Related Item' that refers to a solution artifact which be edited in the IDE, then you can open it with the 'Open Solution Item' menu, or you can double-click on them in Solution Builder.

Viewing Related Guidance

If the solution element has a 'Related Item' that refers to guidance (called a 'Guidance Link'), then you can view the guidance by selecting the 'Open Guidance' menu.

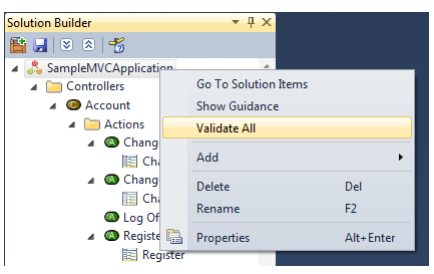


Validating Solution Elements

If the solution element has properties that are required by the pattern, or the values that must have certain values, then validation rules will notify you if these properties and solution elements are invalid, when validated.

When Validation occurs is controlled by the pattern toolkit. Typically, it may occur on any one, or all, of these events occur:

- When any property of any solution element is modified.
- When the solution is built.
- From the 'Validate All' menu on a solution element.
- Other custom events that the toolkit implements.



Running Automation

Solution Elements are the physical launch points, and define the scope, for automation that could be provided by pattern toolkit. Typically, automation is contextualized to a specific solution element, and/or to a set of its descendant solution elements. But some automation may also traverse up and down other solution elements in the pattern model.

The real power of the automation capability in pattern toolkits is that it can be used to do automate just about any task inside of (or even outside of) the Visual Studio environment. Basically, whatever the pattern toolkit needs to do to make development of the pattern quick, easy, and not tedious.

Automation can be triggered by any number of events (e.g. some provided ones include: On Build, On Save, On Creation, On Property Changed etc.) or under any number of conditions (e.g. a Solution Element's Properties are all Valid, or All Related Items in the Solution are Saved, etc.). These conditions are again all defined and controlled by the pattern toolkit.

Most automation uses the properties and the state of the solution elements as its input (or context). Some automation utilizes the state of the other artifacts and services outside the development environment.

It is very difficult to quantify or summarize all the possible types of automation that can be implemented by a pattern toolkit. The scope, depth and quality of the provided automation from a toolkit are entirely dependent on the pattern being modeled, and the other tools available to the pattern toolkit to automate.

There is a provided set of automation types and an automation framework that comes standard with a toolkit, making the application of automation very easy for a pattern toolkit author.

Some general examples of typical kinds of automation in a toolkit would include:

- Generating projects or source code files into 'Solution Explorer' (either from a text transformation or from VS project or item templates).
- Configuring the properties of a related project in the solution.
- Running validation rules when the solution element reaches a correctly configured state.
- Automatically generate reports for traceability.

Concepts

What are Patterns?

Answer. "A Pattern is simply design or an implementation pattern for implementing best practices in IT development and deployment. A pattern is not without some conceptual structure, an implementation, and some parameters."

When implementing complex IT solutions, irrespective of domain, technology or platform selections, it is natural to engineer 'patterns' that abstract implementation detail and focus solely on the concerns being addressed by the creation of the pattern. These patterns are often parameterized so the users of them (those who actually implement and deploy the solutions) can focus on programming them to suit their needs in use. Typically, these kinds of patterns manifest themselves in a set of API's that work together to solve a particular problem. Often, these patterns are formalized and documented. Often they are generalized to adapt to multiple contexts in use.

Patterns are most useful when they can be applied to a well-defined set of use cases or scenarios, and reused more than once.

Problem with reusing patterns include:

- Being able to constrain when they are used, and the context in which they are used.
- Being able to ensure that the parameters of the patterns are configured correctly in use.
- Packaging them with all their dependencies for reuse.

Unless you enjoy writing (and presumably can afford to re-write) every component of your software from scratch, patterns are the most common reuse mechanism in software development today. They tend to be platform and language agnostic, but in that, tend also to have multiple possible implementations. Hundreds of hits on the web for any software development topic are evidence of this. One thing that tends to transcends patterns are best practices, but in order to be effective and efficient and practical all best practice implementation require some form of specificity of platform, language and technology. The end result is an implementation pattern that declares its platforms, languages and technologies. If you can specify those aspects of a best practice pattern, and a consumer can live within them, then the benefits of reuse become significant.

Patterns can have unlimited scope. That is to say you can define a pattern from the smallest thing, like a line of code, to whole systems of software. But in general, larger patterns for complex software solution will almost certainly require narrower cases of reuse and smaller sets of parameters to be effective. As implementation scope increases, practical reuse tends to decrease. Keeping scope and parameters to a practical minimum aids in wider and more frequent reuse. One way to help manage larger scope is to break the pattern into smaller patterns that can be composed into larger software components and systems.

What are Pattern Toolkits?

Answer. "A Pattern Toolkit is a Visual Studio extension that contains templates, automation and guidance that assist you in developing and deploying your solution."

A Pattern Toolkit (like all Visual Studio extensions), integrates with the development environment, providing tools that automate the creation and modification of files, projects and solutions. They can also integrate with the other services provided offered by the development environment.

A Pattern Toolkit is self-contained, and deployed as a *.vsix files that install things like: editors, designers, menus, commands, tool windows, etc.

Visual Studio extensions are managed in Visual Studio using the 'Extension Manager'. For more details about Visual Studio Extensions, see Customizing, Automating and Extending the Development Environment.

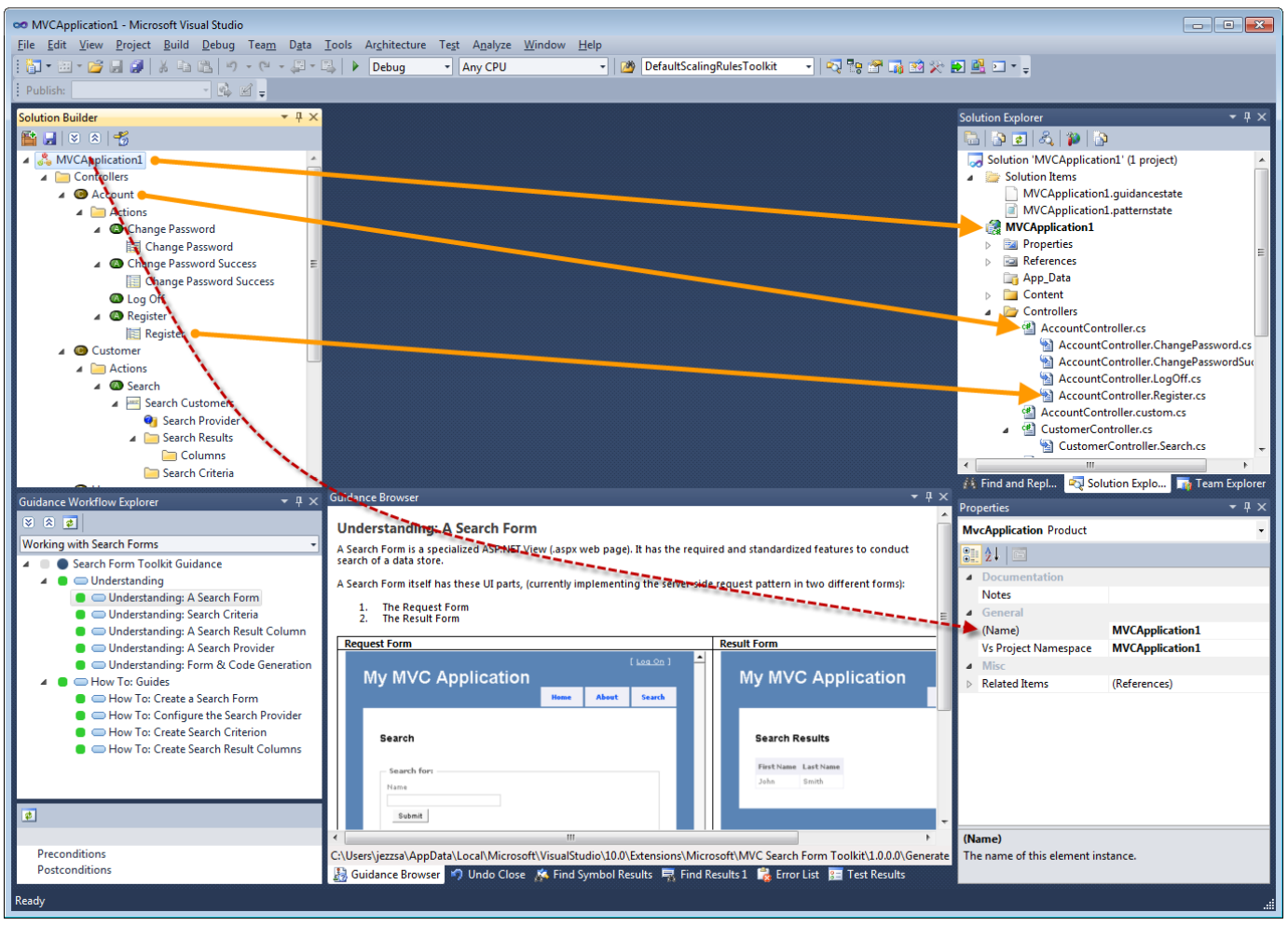
Pattern Toolkits are created using NuPattern. See http://nupattern.org for more details.

What is a Solution Element?

Answer. "A 'Solution Element' is the representation/abstraction of one or more artifacts in the scope of the current solution."

A 'Solution Element' can represent any architectural concept or physical implementation or data structure, and in some cases any composition or arrangement of additional artifacts that can be reached within or outside of the development environment. For example, in any given pattern toolkit, a solution element could represent one or more projects, files, database records, web service responses, configuration data, etc.

A 'Solution Element' can have no physical representation, and just be notional. Its presence in the pattern could be simply structural or conceptual such as for containment or grouping of other solution elements. There are no bounds to what the solution element represents, large or small, abstract or concrete.



A user of a Pattern Toolkit, and the toolkit's built-in automation, work with defined properties of the solution elements to establish their present configuration and state. When verified as valid, the configured state is then used typically to automate and transform that state into a representative implementation in the solution. The sum of the state of all solution elements from a specific pattern toolkit represents the configuration of that instance of the pattern in the current solution.

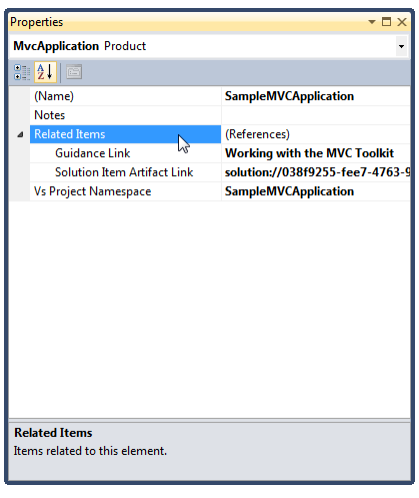
> **Note**: The state is managed by 'Solution Builder', and persisted in a source controlled file in the solution (*.slnbldr).

What are Related Items?

Related Items are references to items or data that do not exist within the pattern model. These items can be are related to, or represented by, the elements of the pattern model. Typically for tracking purposes and for automation. (i.e. the physical source files that are generated from a solution element or source data database records containing data represented by the pattern, etc.).

> **Note**: Related item may exist outside of the Visual Studio environment, such as messages from remote web services, or records in a remote database.

Every solution element has a configurable collection of 'Related Items' and each of these items can be of a different type.



Related Items are a very flexible mechanism to allow the pattern toolkits to integrate with other systems or other tools in the Visual Studio environment to expedite automated development of the solution.

> **Note**: Related items which exist in the 'Solution Explorer' window are commonly referred to as 'artifact links'.

Guidance

What is Guidance?

Guidance is a new capability in general development that provides contextual information about what it is you are developing and how to develop it in a form that includes text, images and other media, to give assistance in developing software in Visual Studio.

As distinct from general Help reference topics on general development or technology topics, guidance is intended to impart expert domain knowledge with recommended practices and processes on how to be productive for the specific solution being built.

Typical guidance, like the guidance you are reading, includes:

- Overview information on the architecture of the solution being built and technologies used.
- Explanations of why and where that architecture is used.
- How and when to use that architecture for the specific solution being built
- Instructions on how to use the purpose built tooling to be productive with this architecture.
- References on how to troubleshoot, test and integrate the software produced as a result.

Although, much of this kind of information can be found from other sources either externally from the public domain, or internally within institutionalized knowledge in a parent organization, guidance is meant to be the condensed collection of that knowledge applied to specific development domain and further contextualized to the current solution being built. Guidance, is not just a high level description of what is to be built, it also contains how to build, test, deploy and troubleshoot it.

In the context of using a Pattern Toolkit, guidance is associated with individual solution elements being developed in 'Solution Builder'.

Guidance is viewed in the 'Guidance Explorer' tool window in Visual Studio, where the user can browse topics, and complete instruction lists that help them be most productive with understanding and using the toolkits in the current solution. That guidance can have state in the current solution, so that users on the same team can pick up work from where other users have left off. Guidance can also track and maintain the state of the solution being built, so that instead of just instructing a user in text what to do next, the guidance can contain hyperlinks that actually execute the tools to do it for them. In this way, working through prescriptive instructional guidance can be very productive for the solution development team.

Each Pattern Toolkit being built delivers its own set of guidance to help users use it patterns and automation to get their work done more productively.

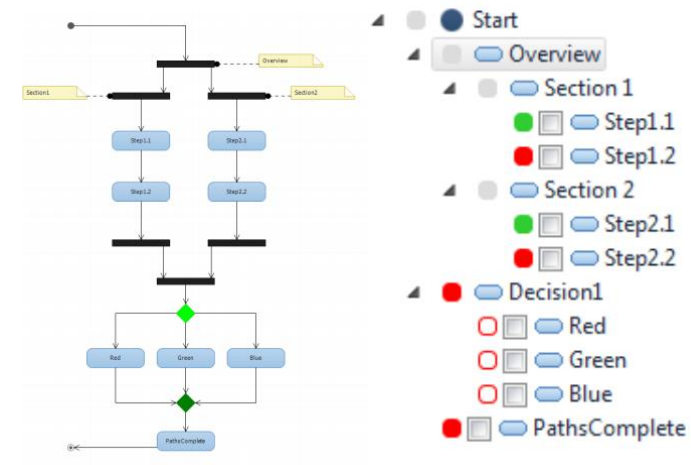
That guidance is composed of one or more a 'Guidance Workflows', each of which is a multi-pathed workflow of individual 'Guidance Documents' (topics), each of which has some browsable content, and each of which may or may not have state to indicate progress through it, and also contain links that automate some part of the described process.

What is a Guidance Workflow?

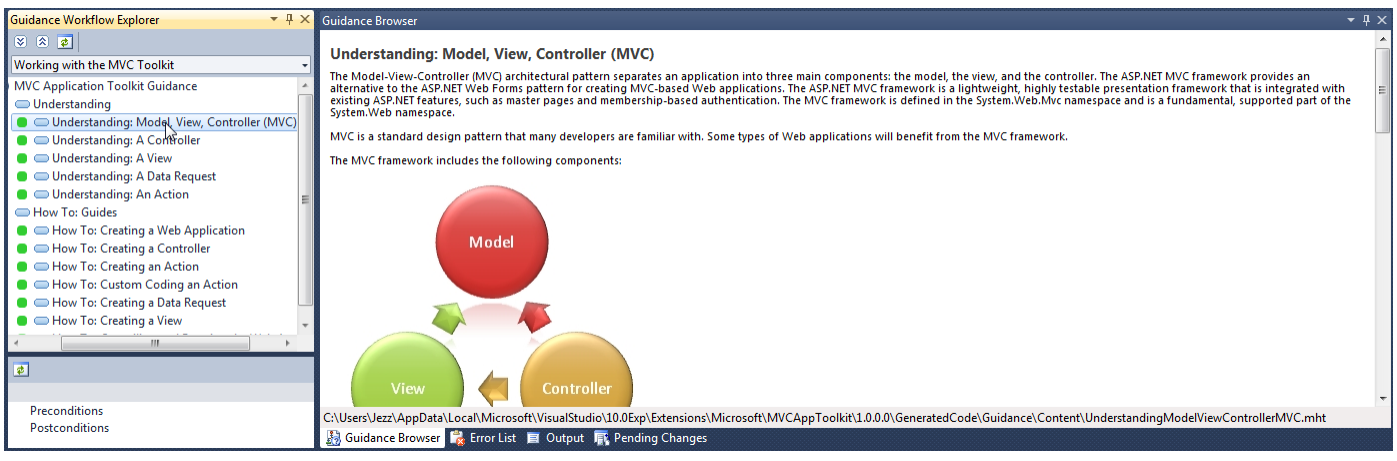
A Guidance Workflow describes the structure (layout) of a set of guidance, that may include both informative and instruction guidance steps.

A guidance workflow operates in much the same way as a classic state-full workflow, in that each step or 'activity' may have an individual state of either: ready, blocked or complete. When applied to guidance, this state directs readers on what they can do now, and what they can't do yet, resulting in a guided set of instructions.

The state of each activity in the workflow is governed by pre and post conditions on the individual activity that must be satisfied to move the state from blocked to ready to complete. The reader can either manually complete the step (ticking a checkbox), or automation can calculate the completion of the step using these pre/post conditions (or some combination of manual and automation). Each activity has a set of name/value pair properties that can also be used to hold state that can be used in the calculations as well.



Each activity may also have associated to it browsable content that can be viewed in the 'Guidance Browser' window when the activity is selected. This content can contain links that are active based upon the state of the activity, and can execute automation using the state of the workflow, and gathered state of the development environment.



What is a Guidance Document?

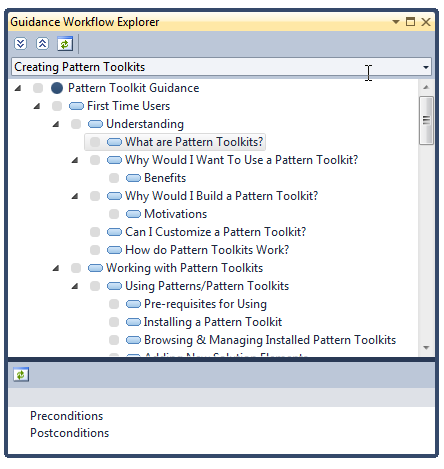
A Guidance Document is simply the content document which can represent a topic of any node in the guidance workflow.

This document is viewed in the 'Guidance Browser' window in Visual Studio. That content can be static content as with MHT page or other textual page, or can be dynamic as with an online web, wiki page or community site page. Ultimately, the guidance document boils down to a URI to some accessible source.

The content of this document is typically some form of HTML that can contain links to other documents within the guidance workflow, or link to external documentation and sites on the web. The content can further contain special automation links (content links) which execute automation commands. These commands can use the state of the current activity in the workflow to configure the command also, so that when a command executes it is contextualized to the work being done in the workflow.

What is the Guidance Explorer?

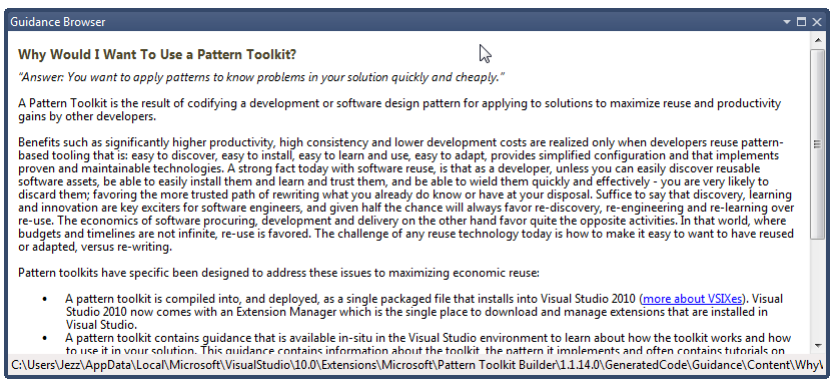
The 'Guidance Explorer' window is the tool window in Visual Studio where you can select, view and interact with the currently active 'Guidance Workflow'.



The Guidance Explorer window works with the 'Guidance Browser' window to show the content of selected topics.

What is the Guidance Browser?

The 'Guidance Browser' tool window in Visual Studio is where you browse and interact with the content of each 'Guidance Document'.



Automation

What is Automation?

Automation is the aspect of a Pattern Toolkit that makes development with the toolkit significantly more productive than hand crafting the software from scratch in a traditional way.

In general, it is the automation that speeds the development with the pattern toolkit by performing activities on behalf of the user that ensure the software being implemented is expeditiously, accurately and consistently implemented.

Without automation, and with guidance, a pattern toolkit could only instruct a user on what to do with its solution-based assets (such as frameworks, libraries and code samples). With automation however, a pattern toolkit can for example: automate the unfolding of project templates, generation of source code, validation of pattern models, support drag and drop, provide contextual menus, run custom tools, refactor code and configuration, synchronize the solution artifacts with the design, etc. There are many areas of a toolkit that leverage automation to make building a solution with them orders of magnitudes faster and cheaper than handcrafting solutions from scratch.

What are Artifact Links?

The term ‘Artifact link’ is commonly used to refer to a ‘Related Item’ of an element that resolves to a solution artifact displayed in the ‘Solution Explorer’ window.

These artifact links provide the ability to track solution items, and are typically used to navigate between the elements in ‘Solution Builder’ window and their associated artifacts in the ‘Solution Explorer’ window. Artifact links are resilient to changes in location in the solution and renaming, since in the development of any solution users constantly move and rename solution artifacts to suit their specific physical structural and naming conventions.

How To: Guides

The ‘How To’ guides provide background information, tips and instructions for performing the most common activities with the Solution Builder tool window.

Using

Installing and developing solutions with pattern toolkits using Solution Builder.

Understanding: What are Pattern Toolkits?

See What are Pattern Toolkits for the concepts.

How To: Install/Uninstall Pattern Toolkits

See Pre-Requisites for Using.

Obtain the Pattern Toolkit Installer

Pattern Toolkits are deployed in a Visual Studio Extension (VSIX) package (a file with the *.vsix extension). A VSIX is a single file, self-installing package containing everything needed to install and run a Visual Studio extension.

You must first obtain a VSIX installer (i.e. MyPatternToolkit.vsix). You can either obtain the *.vsix 'privately' from within your organization, or browse for it 'publically' from the Visual Studio Gallery

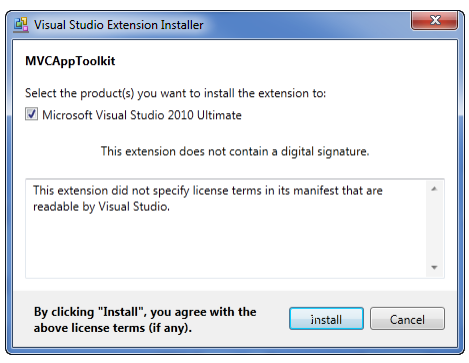
Run the Installer

File Installation

If you have obtained the VSIX file directly, open the file directly from your desktop (hard drive or network location).

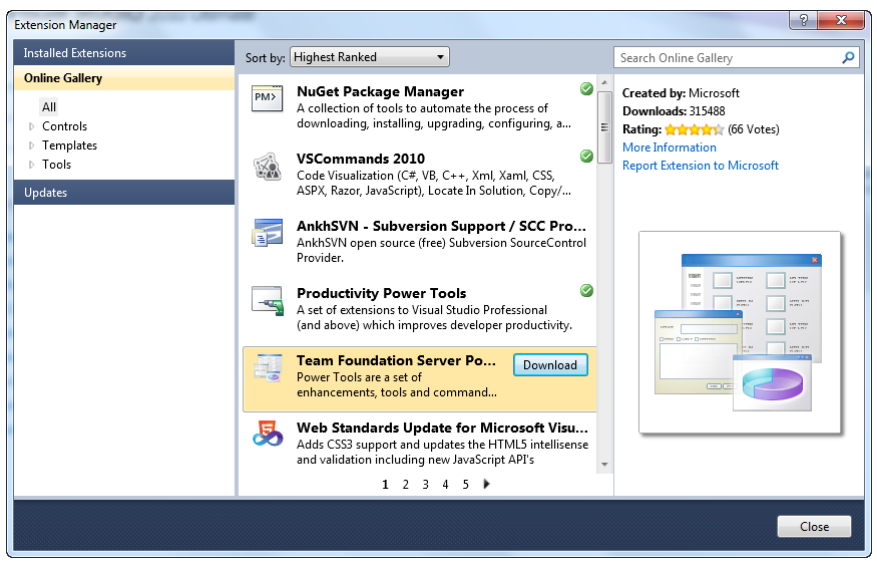
The VSIX Installer will run, to verify your pre-requisites and will install the toolkit into Visual Studio.

> **Note**: Pay close attention to whether the VSIX includes a license agreement, and whether it includes a digital signature to ensure its authenticity and originating source. Both these things are displayed on the front page of the installer when run.



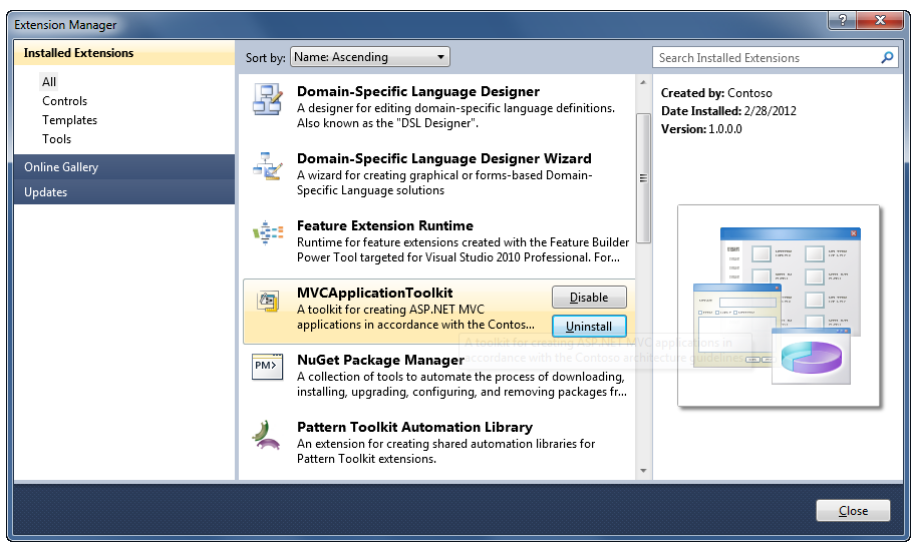
Gallery Installation

If you wish to obtain your VSIX from a gallery, such as the Visual Studio Gallery, then you can search, browse, download and install the VSIX directly from the gallery from the 'Extension Manager' in Visual Studio (Tools | Extension Manager).



Verify Install

Once installed, open Visual Studio, and open the 'Extension Manager' (Tools | Extension Manager) to see your toolkit installed.



Uninstall the Toolkit

You can either 'Uninstall' or just 'Disable' a toolkit directly from the 'Extension Manager'.

> **Note**: Once 'Uninstalled' or 'Disabled', you need to restart Visual Studio to effect the change.

How To: Use a Pattern

> **Important:** In order to use the pattern contained within a Pattern Toolkit, the Pattern Toolkit must be installed. You can verify this by looking at the 'Installed Extensions' in the 'Extension Manager' dialog in Visual Studio (Tools | Extension Manager).

A quick note for understanding; as users, when we say 'use' a pattern (from a pattern toolkit) what we are really doing is using a pattern toolkit to create for us an instance of the pattern that can be parameterized for our solution. This pattern instance is called a 'Solution Element', and we work with it in our solution.

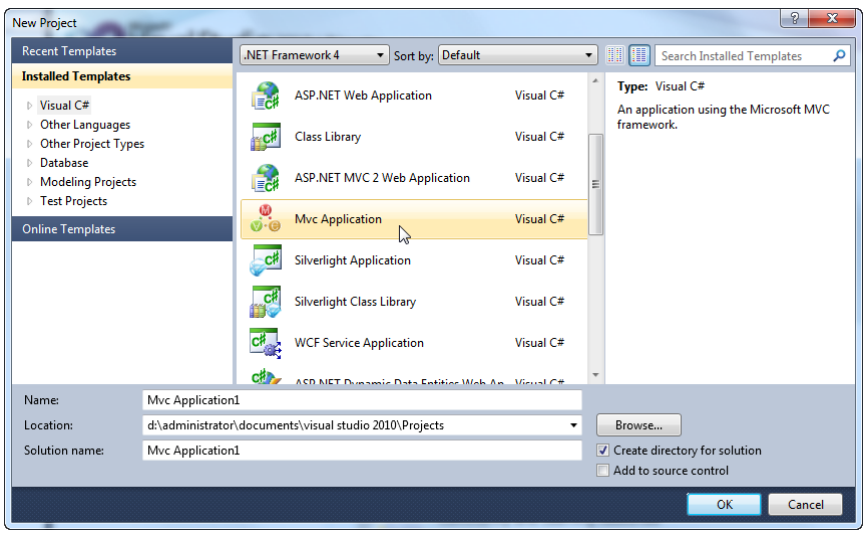
There are two ways to start using the pattern.

Create From the Add/New Project/Item Dialog

If the pattern toolkit includes a project or item template, it will appear in the Add New Project/Item dialog in Visual Studio.

From an empty environment, or existing solution, open the Add/New Project/Item dialog in Visual Studio.

> **Tip**: Use the 'search' box in the top right corner of this dialog to locate the template quicker.



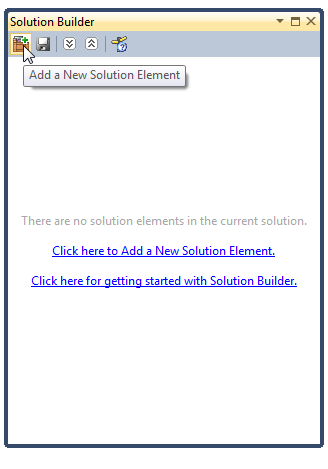
> **Note**: If you cannot find the pattern from this dialog, you can always find it from the next method below

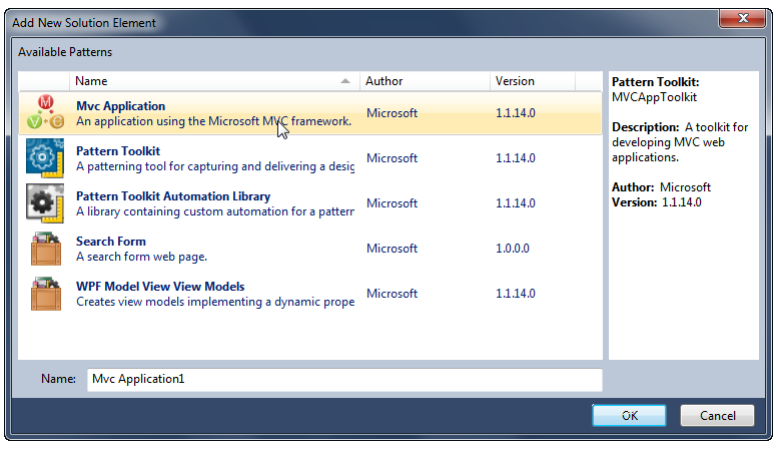
Create From Solution Builder

Otherwise, all patterns can be found in the 'Add New Solution Element' dialog.

> **Note:** You must have an existing solution open in order to create a new pattern this way. Create a new blank solution if you don't already have one open.

From an existing or new solution, open the 'Solution Builder' tool window, and click the 'Add New Solution Element' button (or hyperlink in the window).





> **Note**: Even though a pattern may or may not appear in the Add New Project/Item dialog in Visual Studio (depending on whether it supplies a project or item template), all patterns appear in the 'Add New Solution Element' dialog, in the 'Solution Builder' tool window.

Apply the Pattern

In either case, a new instance of the pattern will be created for you and displayed in 'Solution Builder', and you work with the created 'Solution Elements' in that window with the tools and guidance the pattern toolkit provides for you.

See Viewing and Configuring Solution Elements for examples of the kinds of automation and guidance the toolkit may provide.

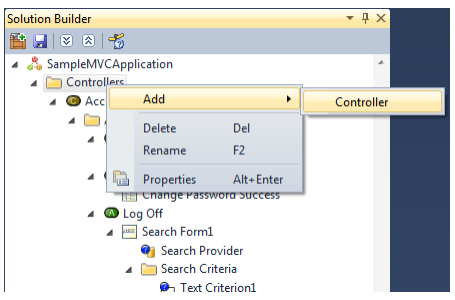
How To: Add New Solution Elements

If a solution element has child elements defined in its pattern, then the 'Add' menu will appear in the context menu of the element.

Add a New Element

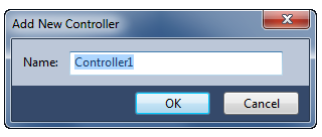
You can add new instances of child elements by right-clicking on the element and selecting the type of child to add from the 'Add' menu.

> **Note**: Some elements in a pattern may only allow one instance of that element, and in those cases, the menu may be disabled when one instance is already present.



Name the New Element

Give the new element a name.



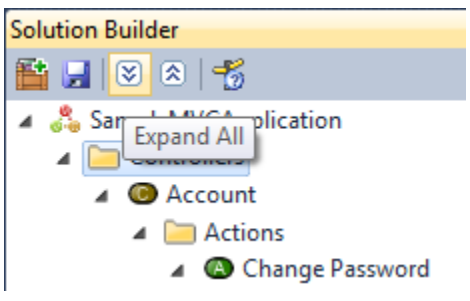
> **Note**: If any automation executes as a result of creating this new element, then you may see activity in the development environment as this element is created.

How To: Control the display of Solution Elements

In 'Solution Builder' you can control how solution elements are presented in the tree layout:

Expand/Collapse

The Expand/Collapse buttons apply to all elements in the Solution Builder.

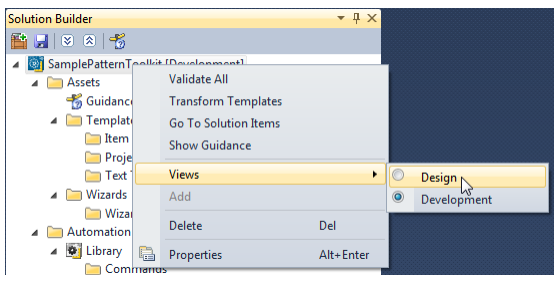


How To: Work with Multiple Views

If a pattern supports multiple views (or aspects), then you can switch between these views in 'Solution Builder'.

Patterns will provide multiple views to focus development on different distinct aspects of the pattern.

> **Note**: Every pattern has at least one view, the 'default' view, which is the current view when the solution is opened.
> If a pattern only has one view, then the 'Views' menu is not displayed.

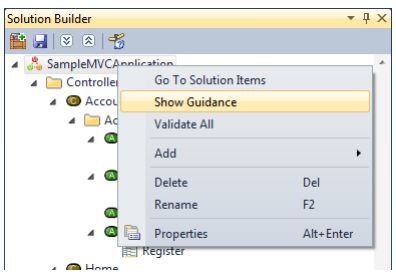


How To: Find the Guidance

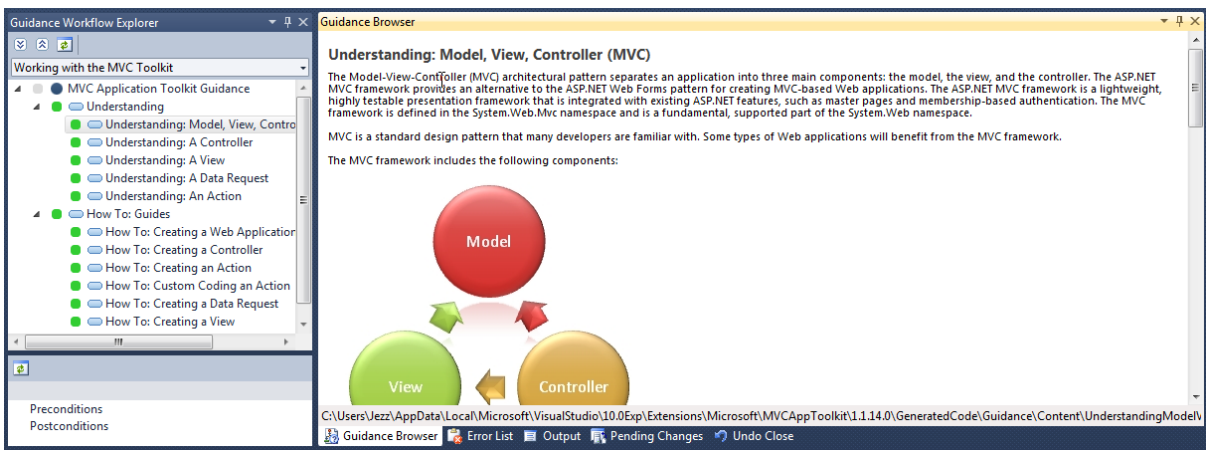
If a solution element is associated to any guidance, then you can locate and browse that guidance from the element.

Show the Guidance

Right-click on the element and select the 'Show Guidance' menu.



The guidance then appears in the 'Guidance Explorer' and 'Guidance Browser' tool windows.



Note: It is not common for this menu to be available for all elements in the pattern. Typically, it is only provided for the top level element of the pattern, or for key child elements that have important distinct processes related to them.

How To: Run Automation

In most cases, automation automatically runs for you when working with solution elements in Solution Builder. The toolkit providing the pattern takes care of this for you. This is the whole benefit of having toolkits implement solutions for you based on patterns.

For example, you may see new projects and files created or updated in 'Solution Explorer' as you add or change solution elements. You may see errors appear in the 'Error List' window informing you when things aren't configured correctly, or when critical steps have not been followed.

Expect that this kind of automation may be automatically performed when: new elements are created or their properties changed, or when the solution is built, or when saved etc.

Some automation needs to be executed manually.

Some automation may require special conditions to be satisfied before it can execute. Code generation is typical example of such automation.

In any case, automation may be conditional on a number of things. Such as: the state of the solution elements in 'Solution Builder', or the state of related projects and files in 'Solution Explorer', or maybe even the status of other systems and services inside or outside the development environment.

Executing Automation Manually

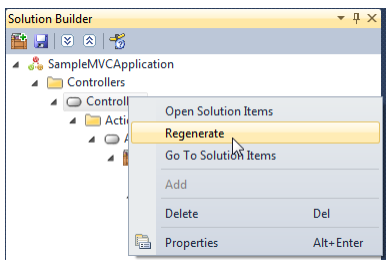
Depending on how, where and when the specific toolkit chooses to apply automation in the application of the pattern, it may need to be executed manually at certain times.

Sometimes, this automation is critical to the patterns development, sometimes they are ancillary. For example, an 'Export' function may not be necessary to get the pattern completed in the solution, although very useful for extracting or persisting some data as a result of applying the pattern to the solution.

The toolkit may guide you to when these things are ready for executing, either with guidance, or perhaps with errors in the 'Error List' to prompt you to complete a task. Sometimes, it is apparent from the pattern you are applying.

Manual automation is typically run from content menu items on solution elements in Solution Builder.

Simply click the menu item to execute the function, and the toolkit will manage the rest.

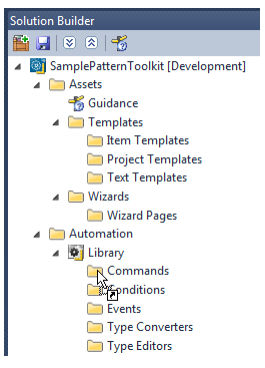


How To: Use Drag and Drop

It is common for solution elements in 'Solution Builder' to allow drag and drop of data from other windows (i.e. 'Solution Explorer').

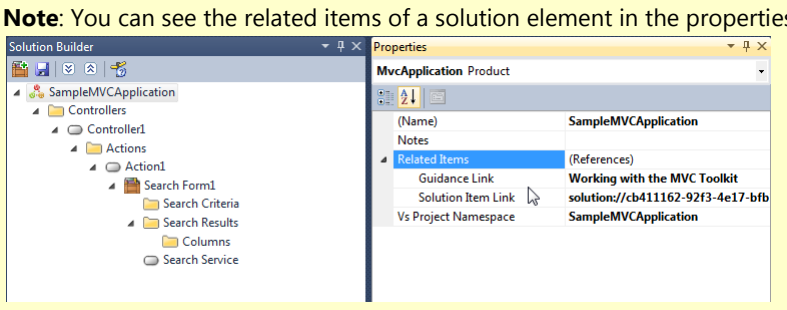
The source for the data can be anything, e.g. files from a hard drive, items in a solution, records from a database etc.

Simply drag data over solution elements in the Solution Builder window, and if the mouse allows, drop the data on the solution element.



How To: Navigate to or Open Solution Items

It is common for solution elements in 'Solution Builder' to be related to one or more solution items displayed in 'Solution Explorer'.

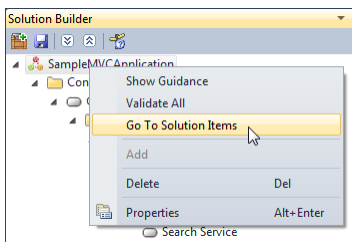
> **Note**: You can see the related items of a solution element in the properties of the solution element.
>
> 
>

For these solution elements, you will be able to navigate from elements in 'Solution Builder' to project items in 'Solution Explorer'.

> **Note**: Some solution items cannot be opened in any editor, such as: the solution, any folders and projects. In these cases you will only be able to navigate to these solution items, which simply selects them, and displays them in 'Solution Explorer'.

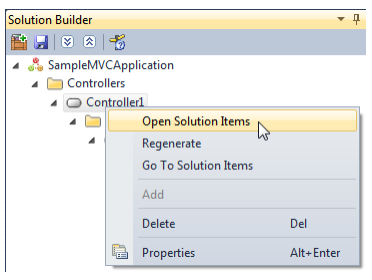
Navigating to Related Artifacts

To select and navigate to the solution items in 'Solution Explorer', right-click and select the 'Go To Solution Items' menu.



Opening Related Artifacts

To open the solution items in the development environment, right-click and select the 'Open Solution Items' menu, or double-click on the solution element.



Fixing Invalid Related Artifacts

If the related item cannot be found, or does not exist in the solution, then the related solution items may have been moved, renamed or deleted from the solution.

You can fix the link to the item. See Fix Related Items for details.

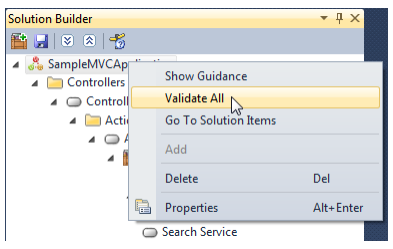
How To: Validate Solution Elements

Use validation to verify that the solution elements, and therefore your application of the pattern, is correctly configured in 'Solution Builder'.

Depending of the pattern toolkit you have installed, validation may occur automatically as you create and configure solution elements (i.e. when a property changes, or when solution built), or you are given menu commands to execute validation manually. Some toolkits offer both.

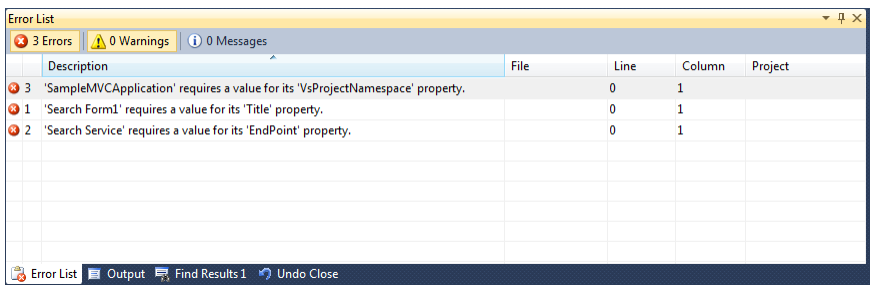
Validating Elements

Typically, menu items like the 'Validate All' menu are provided on the top level solution elements.



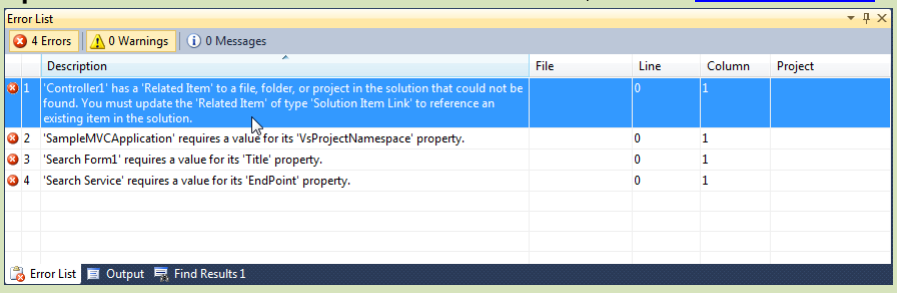
Validation Errors

Validation errors and warnings are reported in the 'Error List' tool window (CTRL + W, E).



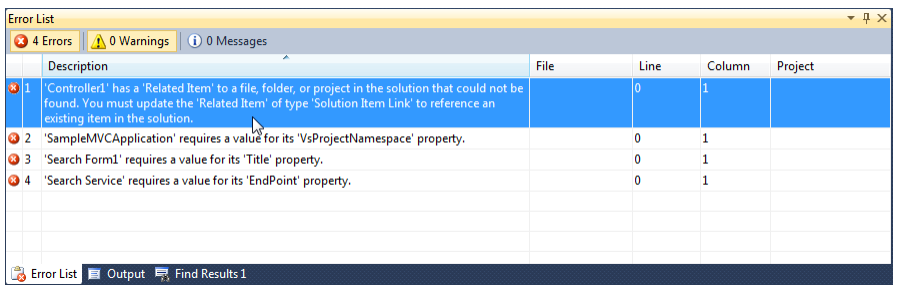
Note: Double-clicking on a validation error does not currently navigate you to the offending element in 'Solution Builder'.

Tip: If a validation error refers to broken 'Related Items', then see Fix Related Items.



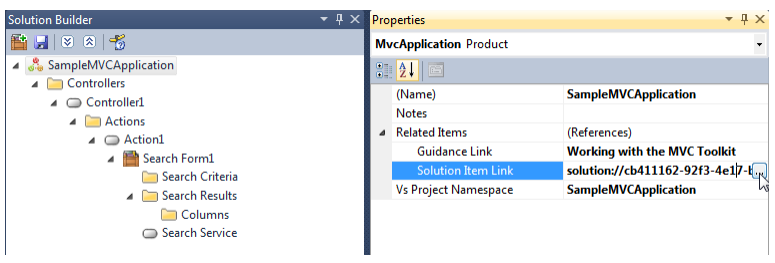
How To: Fix Related Items

If a related item of a solution element cannot be found in the current solution, then an error appears in the 'Error List' window similar to this:



There are many types of related items for a solution element. They may refer to items in the Solution Explorer, or in fact any other source in Visual Studio. Some related items may even refer to sources and services outside the Visual Studio environment.

Depending on the type of Related Item, the value of that item in the 'Properties Window' will provide the necessary editors to help re-select or define a new value.



For the 'Solution Item Link' kind of related item, a solution picker helps you select the correct file, folder or project to relate to.

How To: Troubleshoot Pattern Problems

Occasionally, problems with specific toolkits may occur. Some problems are known temporary glitches and can be worked around, others may result in blocking issues that prevent further toolkit usage.

Diagnosing and Reporting Issues

Either way, these problems need to be identified and reported to the toolkit author by using their provided feedback mechanism.

See Troubleshooting Toolkits for more details on these processes.

Authoring

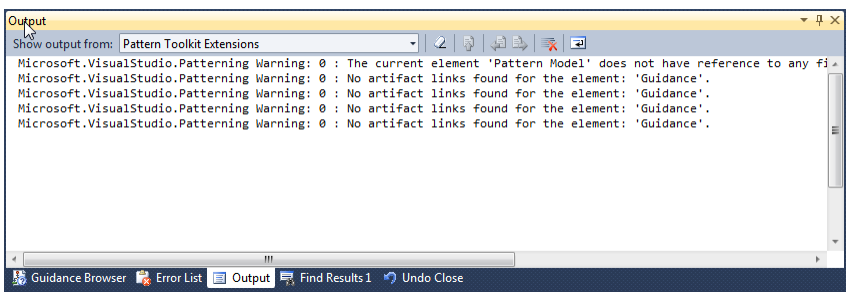
For ‘How To’ style guidance for authoring pattern toolkits, you must have the ‘Pattern Toolkit Builder VS2010’ or ‘Pattern Toolkit Builder VS2012’ (VS012) extension installed in Visual Studio, and you must create a new ‘Pattern Toolkit’ project in Visual Studio to get access the guidance.

Reference

Troubleshooting Toolkits

If errors or inconsistencies are experienced with building or using pattern toolkits, you can find out more information about what went wrong by looking at the diagnostic trace information displayed by the 'Output Window' of Visual Studio, in the 'NuPattern Toolkit Extensions' pane.

See the Tracing Window for more details.



This kind of detailed information is useful for identifying the causes and the data leading up to the issue.

Diagnosing Issues

By default only 'Warning' and 'Error' diagnostic level traces are captured in the trace window, but you can increase/decrease the fidelity of the information by changing the tracing levels up to 'Information' or 'Verbose', or down to 'Warnings' or 'Errors'.

You change the level of diagnostic information displayed by changing the trace level in the 'NuPattern Toolkit Extensions' Options.

> **Note**: Changing the tracing level only affects new traces, so if troubleshooting a specific issue, you will need to clear the trace window and reproduce the issue again to see the new level of trace messages.

Reporting Issues

Issues with toolkits that cannot be worked around will need to be reported back to the authors of the toolkits being used.

> **Note**: The provided feedback mechanism is specific to each toolkit.
> In general, you should refer to any troubleshooting type sections of guidance provided by the specific toolkit for details about how to contact toolkit owners and report issues.

When reporting any problem try to give as much detail on what problem you are seeing and what set of actions led up to the problem. Including detailed trace logs, that is, with the trace level or at least 'Information' is most desirable.

> **WARNING**: Detailed trace logs, depending on the toolkit, may contain sensitive information. Please ensure this information is sanitized before copying or sending.

Screen shots also help a great deal, short lists of the actions you took to reproduce the problem and more detailed information gathered from trace logs also helps to identify the root cause of the issues.

Resetting the Environment

If you are building a pattern toolkit, and are using the Experimental Instance of Visual Studio to test your toolkit, you may also try resetting the Experimental Instance it to clear out any remnant or duplicate data or settings which could be causing issues.

Environment

The development and tooling environment for using, authoring and customizing Pattern Toolkits.

Visual Studio Experimental Instance

The 'Experimental Instance of Visual Studio', is a special testing version of Visual Studio that is primarily used to test and debug Visual Studio extensions under development. As opposed to the 'Normal' instance of Visual Studio where regular code development takes place.

> **Note**: All the settings needed for Visual Studio are kept in the registry, and the experimental instance using a different set of registry settings than the normal instance of Visual Studio.

This special experimental instance of Visual Studio can be reset at any time, to clear out any detritus from testing or developing extensions against it, without affecting the 'Normal' instance of Visual Studio where you do your regular development.

Running Experimental Visual Studio

To run to the 'Experimental Instance', run the **"Start Experimental Instance of Microsoft Visual Studio 201X"**, from the Start Menu (All Programs | Microsoft Visual Studio 201X SDK | Tools).

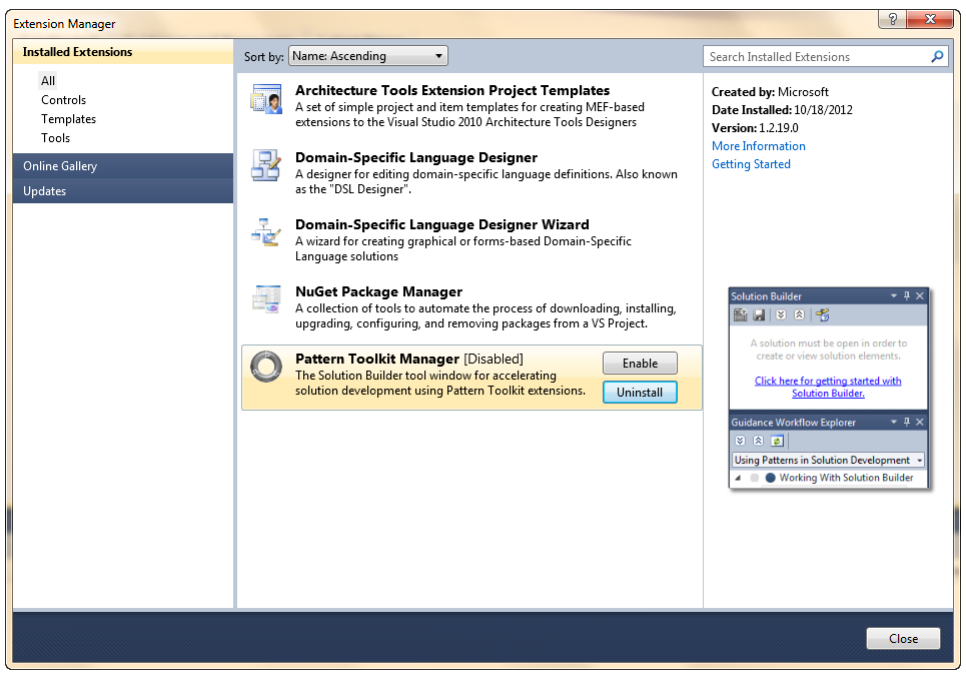
Resetting Experimental Instance

To reset the Experimental Instance of Visual Studio, (specifically for pattern toolkit development) requires these steps:

1. Close all running instances of Visual Studio 201X.
2. Run the **"Reset the Microsoft Visual Studio 201X Experimental instance"**, from the Start Menu (All Programs | Microsoft Visual Studio 201X SDK | Tools).

> **WARNING**: In some cases this command prompt fails to run correctly, and does not reset the experiment hive.
> The symptom of this is if the command window flashes (immediately opens and closes), and does not ask you to "Press any key to continue…" after it finishes the process.
> If this occurs, simply delete the directory at: **%localappdata%\Microsoft\VisualStudio\1X.0Exp** and run the command again.

3. Run the **"Start Experimental Instance of Microsoft Visual Studio 201X",** from the Start Menu (All Programs | Microsoft Visual Studio 201X SDK | Tools).
4. Open the 'Extension Manager' dialog (Tools | Extension Manager…), and enable all the extensions which have the **'[Disabled]'** status.

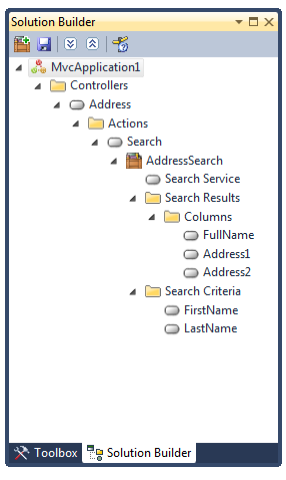


> **Note**: If you see any of your toolkits here that are under development at this point, you should probably uninstall them also.

5. Close the 'Experimental Instance' of Visual Studio.
6. Rebuild your toolkit solutions

Solution Builder

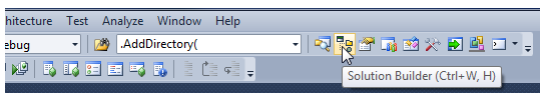
The 'Solution Builder' tool window is a new tool window for working with patterns in your solution.



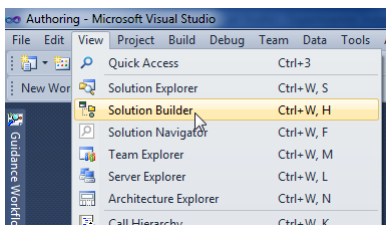
You use this window for creating 'New Solution Elements' that help automate the creation of projects in your solution using installed patterns.

To show the tool window, either:

- Click on the 'Solution Builder' button on the main toolbar in Visual Studio.



- Click on the 'Solution Builder' menu (View menu)



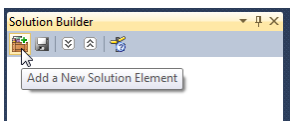
- Press **CTRL + W, H**

> **Tip**: Move and dock the tool window in a separate area of the main window of Visual Studio (i.e. on the opposite side of the window from 'Solution Explorer'), so that you can see both 'Solution Builder' and 'Solution Explorer' at the same time. This is useful for working on the solution using both these windows.

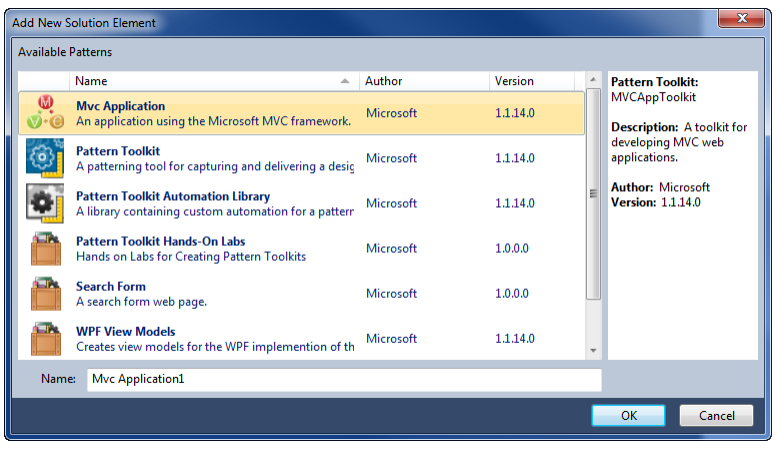
Add New Solution Element Dialog

The 'Add new Solution Element' dialog is where you create new instances of patterns in your solution, these new instances called 'Solution Elements' are then created in the 'Solution Builder' window.

You can open this dialog box from 'Add' button  the 'Solution Builder' window.

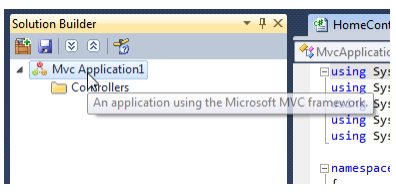


> **Note**: You will need to have an existing or new solution already open to create new solution elements with this dialog.



From this dialog, you select the pattern you want to use, and name it.

A new instance of that pattern, with that name is then created in the 'Solution Builder' window.



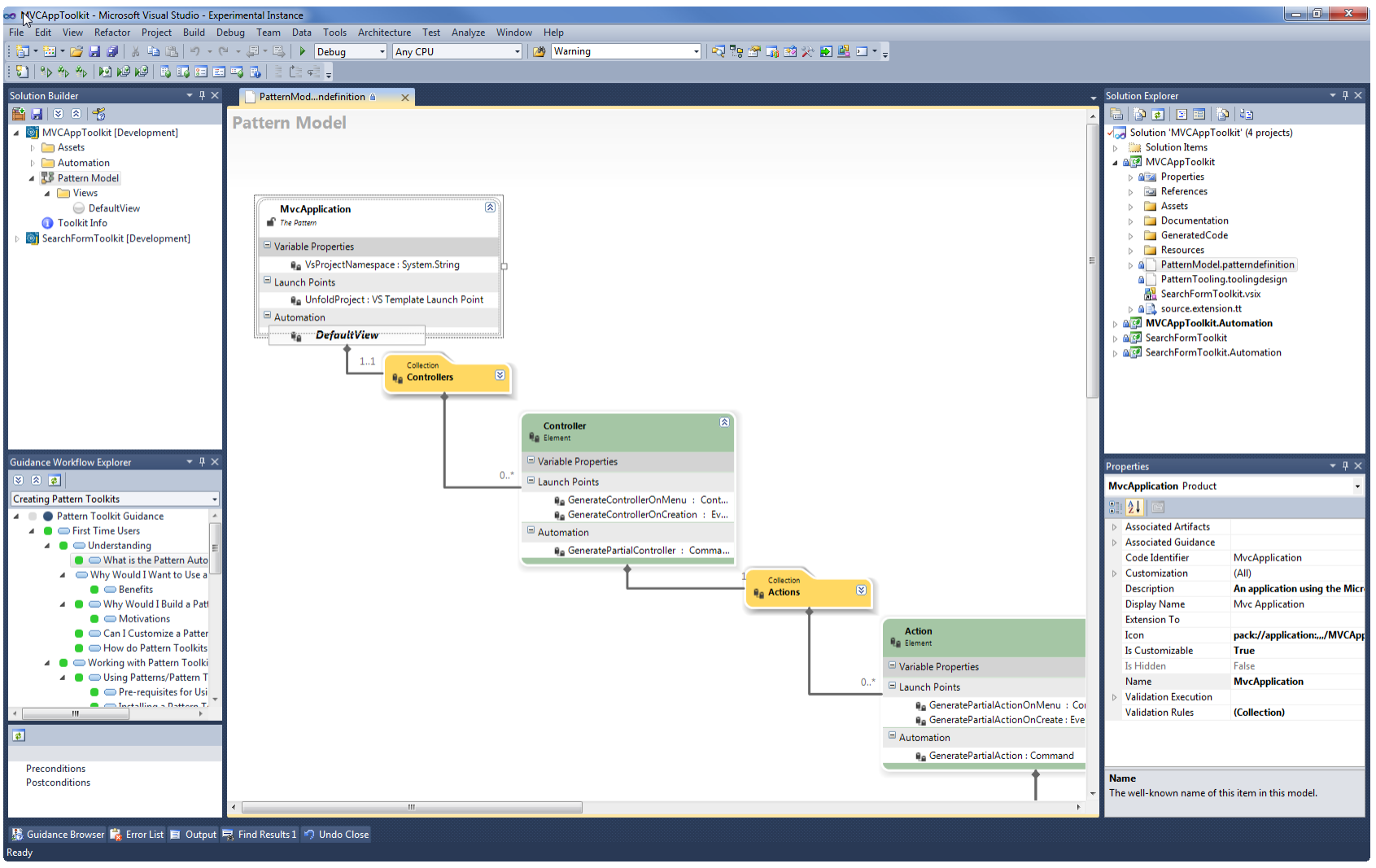
Pattern Model Designer

The 'Pattern Model Designer' is a designer for describing the pattern within a Pattern Toolkit.

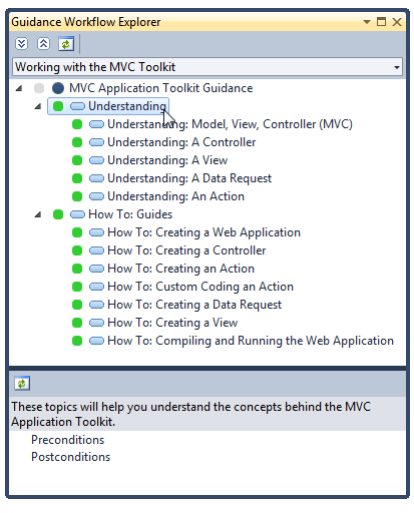
This designer defines the pattern, and the elements which allow a user to configure its variability for their solution.

You open the designer from the 'Pattern Model' element in 'Solution Builder', or the 'PatternModel.patterndefinition' file in 'Solution Explorer'.

You modify the shapes on the designer with the 'Properties Window'.



Guidance Workflow Explorer

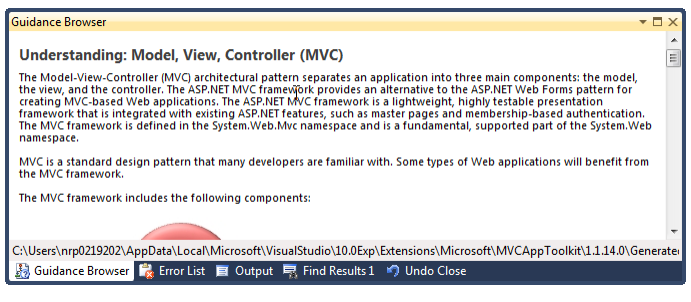


The 'Guidance Workflow Explorer' window (View | Other Windows menu) in Visual Studio displays the available guidance workflows that can be used to work with other tools and processes in Visual Studio.

This window allows you to select a workflow from the drop down list, click on each step in the workflow.

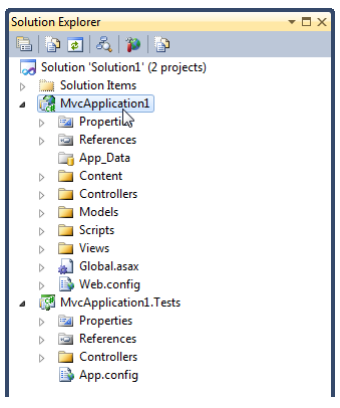
Each step in the guidance workflow, may be either active (green), blocked (red), or disabled (grey) to indicate whether the user can proceed with that step in the guidance. Some steps will include a checkbox that allows the user to mark the step as complete. Other steps may determine if they are complete using automation and conditions.

For steps that have more information, you can select the step in the workflow and browse the guidance in the 'Guidance Browser' window.



More details on 'Guidance Workflows' and how they work will be provided soon.

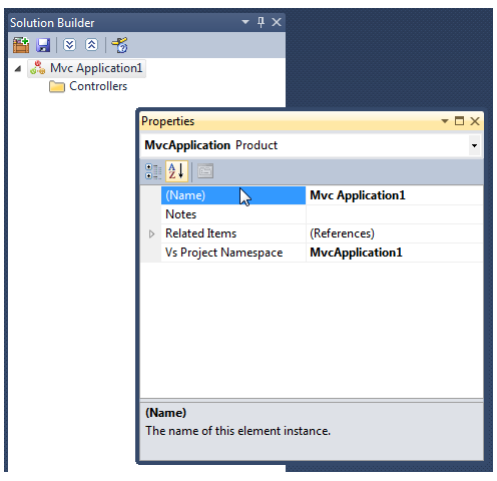
Solution Explorer



'Solution Explorer' (CTRL + W, S) is a common tool window in Visual Studio for displaying the solution and projects under development.

This window shows the physical representation of a solution, with its files, folders and projects.

Properties Window



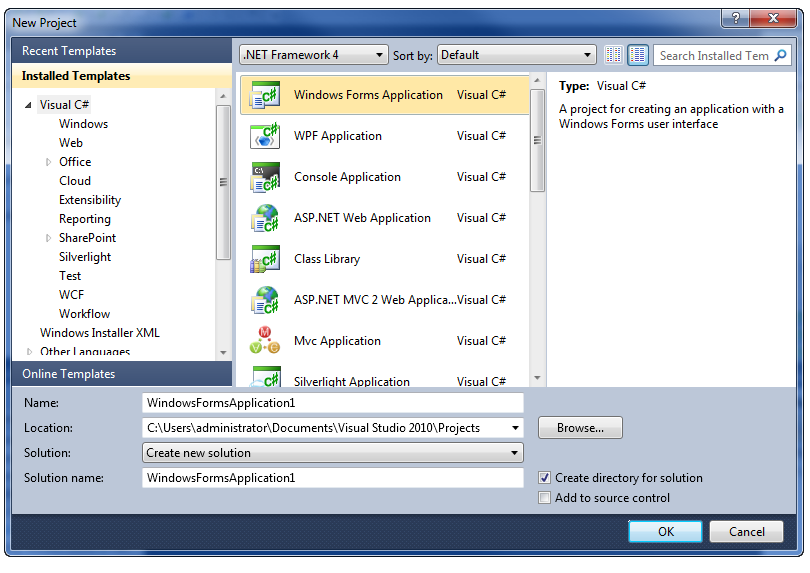
The 'Properties Window' (CTRL + W, P) in a common window in Visual Studio for displaying the properties for the selected object in the currently active window.

In this window you can edit the properties of the selected object, and it provides editors and dialog for manipulating these properties.

Add/New Project/Item Dialog

The Visual Studio 'Add New/Add Existing Project or Item' dialogs (File | New or File | Add menus) are where you add new projects and items to projects in your solution.

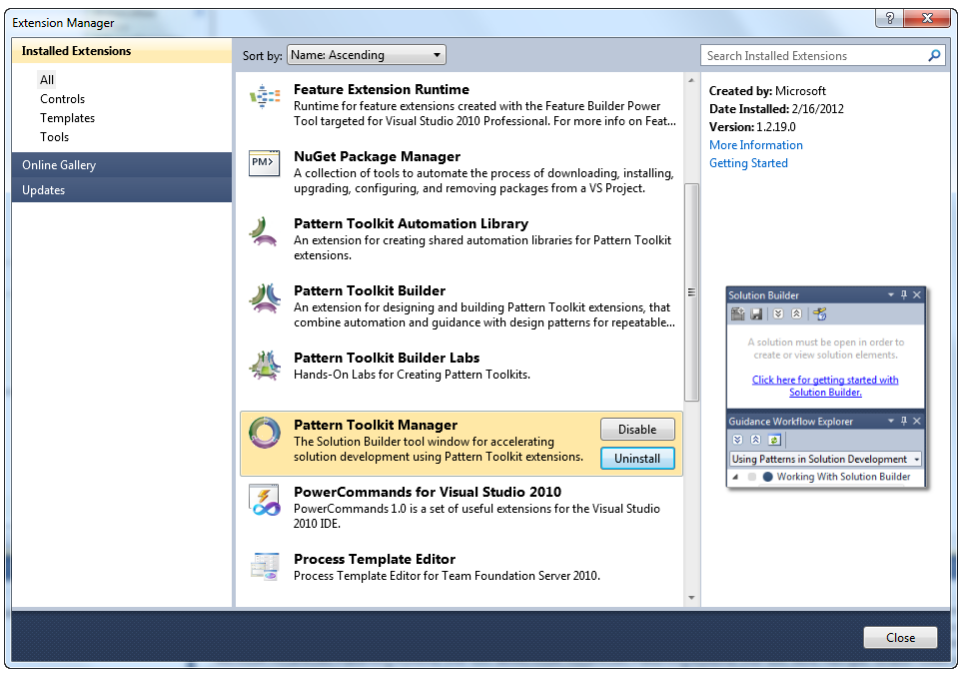
This dialog lists all the project and item templates installed on the current machine.



> **Note**: Item templates are filtered based upon the projects the items will be added to.

Extension Manager

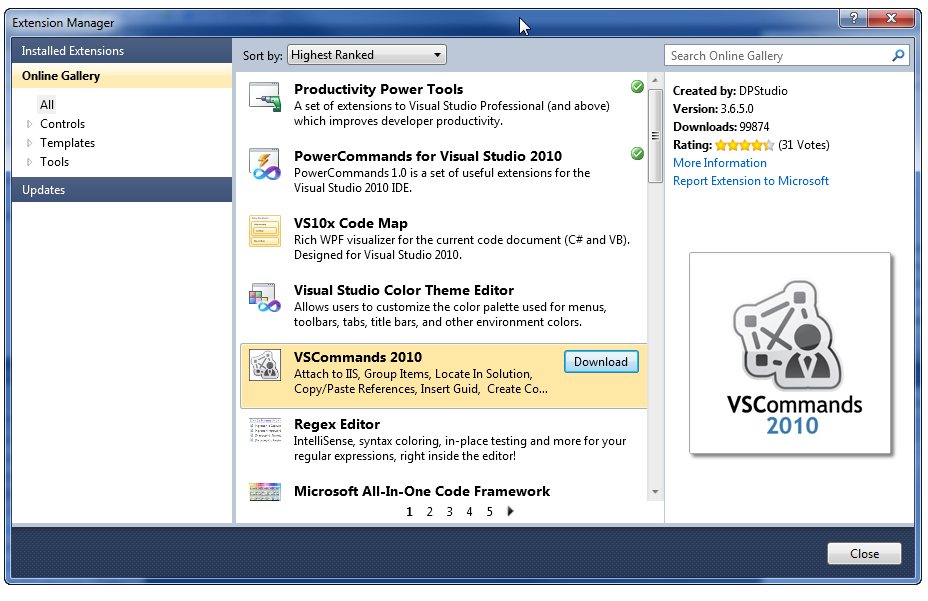
The Visual Studio 'Extension Manager' (Tools | Extension Manager menu) is where you view and manage the installed extensions to Visual Studio.



In the dialog above you can see the currently installed extensions.

> **Note**: NuPattern extensions are listed as extensions in 'Extension Manager'.

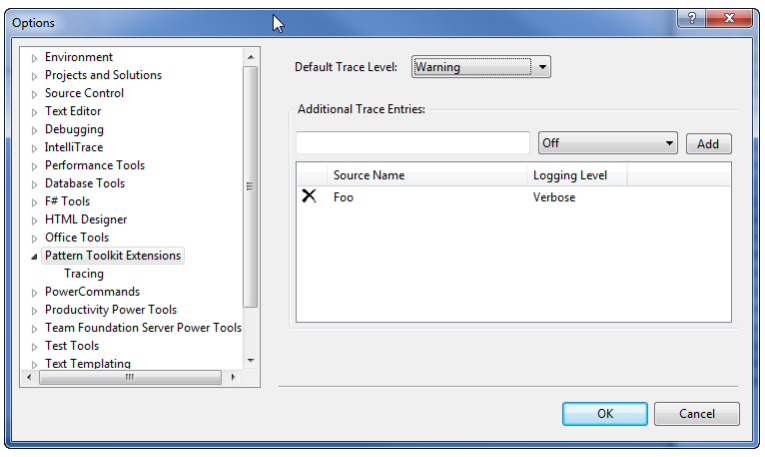
You can also download and install additional extensions from the [Visual Studio Code Gallery](#) in this dialog.



Options

There are several options that control how patterns toolkits work in the Visual Studio environment.

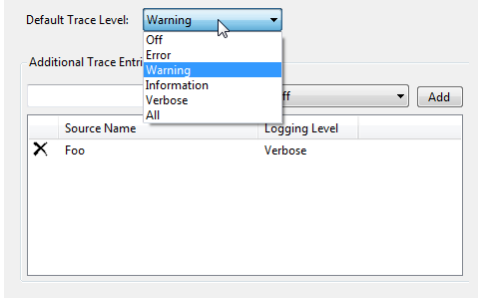
They are modified in the Visual Studio 'Options' dialog, (from the Tools | Options menu)



Tracing Options

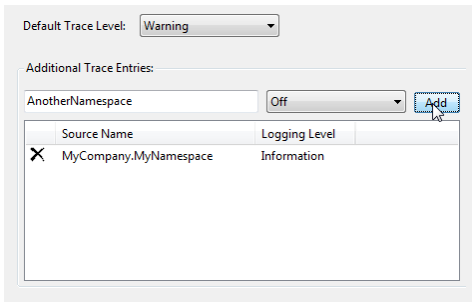
In this page you can control the tracing levels and tracing sources that appear in the Tracing Window.

To change the default trace level for all toolkit diagnostic information, select the 'Default Trace Level'.



> **Note**: The different levels are ordered such that the items lower in the list include all items above them. So for example, by choosing 'Information' you are choosing 'Information', 'Warning' and 'Error' levels.

To add additional trace sources for deeper diagnosis, add a new trace entry to the 'Additional Trace Entries' list.



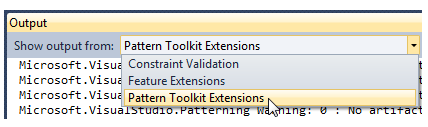
> **Note**: The format of trace sources is typically a name of an application or namespace of the types that produce trace output.
> See Introduction to Instrumentation and Tracing in the .NET Framework.

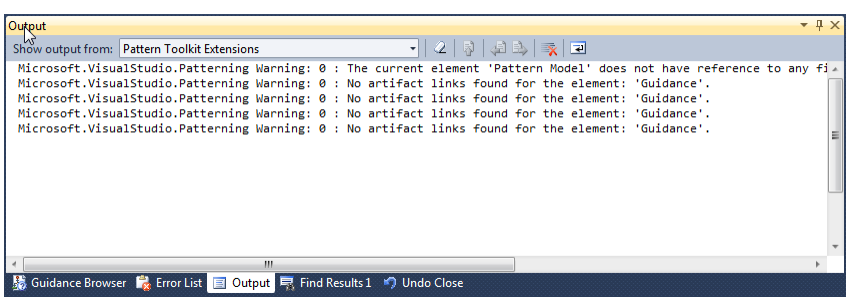
Tracing Window

The Tracing Window is where status and troubleshooting information is displayed for all pattern toolkits.

Opening the Trace Window

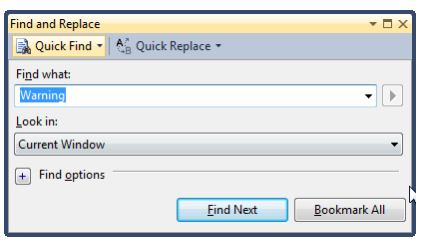
Open the 'Output Window' (CTRL + W, O) in Visual Studio, and select the 'NuPattern Toolkit Extensions' pane from the drop down list at the top.





Working with the Trace Information

The trace information can be searched using 'Find' (CTRL + F), for searching for specific text.



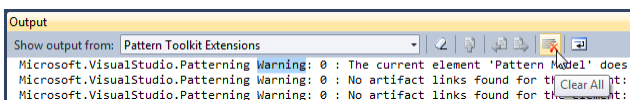
Select the 'Current Window' option to search this information.

> **Note**: Make sure the 'Output Window' is first pinned, and the mouse is first clicked in the window before searching.

Trace information can also be copied to the clipboard.

> **Tip**: This is useful for sending the trace information to others, for example in an error report.

You can also clear the information by clicking the 'Clear All' button in the tool bar of the 'Output Window'.



Adding More Trace Information

You can change the level of diagnostic information displayed in this trace window, or add trace information from other sources by changing the options in the 'NuPattern Toolkit Extensions' Options.

For example, you can increase the level of diagnostic trace information for all toolkits, and add additional trace sources for other extensions running in Visual Studio.

More Information

You can find more information at the NuPattern project site http://nupattern.org

Pattern Toolkits are compiled into Visual Studio extensions which are packaged and deployed as VSIX files.

- What is a VSIX?
- Visual Studio Extension Deployment
- VSIX Deployment

Known Issues

This is a list of the critical known issues in the current version of NuPattern.

Build error: "store must be open for this operation"

Symptoms:
In Visual Studio 2010, when you build a pattern toolkit project you get a build error "store must be open for this operation", and the output window indicates a problem with a MS Build target in the Microsoft.VsSDK.targets file.

Cause:
The version of the Visual Studio 2010 SDK is not compatible with the version of Visual Studio 2010.

It is likely you have Service Pack 1 for Visual Studio 2010 installed, but do not have Service Pack 1 of the Visual Studio 2010 SDK installed.

Solution:
You must uninstall the current version of the VSSDK installed on your machine, then install Service Pack 1 of the Visual Studio 2010 SDK

Feedback

All feedback, bugs, suggestions, questions etc. for NuPattern are very welcome at the NuPattern project site: http://nupattern.org