

MSBuild Sidekick v4.5

User Manual

Contents

1.	What is MSBuild Sidekick?	4
2.	System Requirements	4
3.	Installation Instructions	4
4.	Purchasing MSBuild Sidekick	4
5.	MSBuild Sidekick Features List	4
6.	Graphic User Interface Outline	5
	6.1 Menus	6
	6.2 Toolbars	9
	6.3 Project Tree Window	9
	6.4 Properties Window	11
	6.5 Elements Window	12
	6.6 Raw XML Window	12
	6.7 Help Window	13
	6.8 Targets Diagram Window	13
	6.9 Debug Window	14
	6.10 Logs Window	15
	6.11 Globals window	17
	6.12 Autos window	
7.	How-To Notes	
	7.1 How To Use Tree Modes Effectively	
	7.2 How To Add Tasks To Target	
	7.3 How To Specify Tasks Output Parameters	
	7.4 How To Add New Elements	
	7.5 How To Use Raw XML	
	7.6 How To Use Conditions Edit Shortcuts	20
	7.7 How To Navigate Around	20
	7.8 How To Rename Elements	21
	7.9 How To Delete Elements	
	7.10 How To Use Build Options.	
	7.11 How To Use Target Diagram.	
	7.12 How To Debug Step-by-Step.	
	7.13 How To See Property Values When Debugging	
	7.14 How To Use MSBuild Extension Pack Tasks	
8.	Configuration	
	8.1 Application Configuration	
	8.2 Project Configuration	
9.	Advanced topics	32
	9.1 Solution files	32
	9.2 Override Target	
	9.2 Custom imports	
	9.3 Custom using tasks	
	9.4 Loggers	
10). MSBuild 2.0. 3.5. 4.0. 4.5 support	
	10.1 .NET Framework Version	
	10.2 MSBuild Schema Version Configuration	
	10.3 Toolset Version Configuration	
	10.4 MSBuild 3.5 Editor Support	
	10.5 MSBuild 4.0 Editor Support	
	10.6 MSBuild 4.5 Editor Support	36
	Toto his ballot oupport	

11.	Application Update	
12.	Walkthroughs	
11.	1 Walkthrough: Creating MSBuild project from scratch	
11.	2 Walkthrough: Building and using build logs	44
11.	3 Walkthrough: Debugging and using Globals/Autos windows	48
11.	4 Walkthrough: Building Visual Studio C# project different configurations	52

1. What is MSBuild Sidekick?

MSBuild Sidekick offers easy-to-use Graphic User Interface for authoring, building and debugging project files for Microsoft[®] Build (MSBuild) Engine.

MSbuild Sidekick has built-in support for tasks provided as part of <u>MSBuild Extension Pack</u>, providing user-friendly editors and online help for these tasks.

MSBuild Sidekick can be used with any XML file conforming to MSBuild 2.0, 3.5, 4.0 and 4.5 schema (see also System Requirements below), including but not limited to

- Generic MSBuild projects (*.proj, *.targets, *.msbuild, *.tasks)
- Microsoft Visual Studio 2005/2008 C# and VB .NET projects (*.csproj, *.vbproj)
- Microsoft Visual Studio 2010/2012 all types of projects
- Borland (aka CodeGear) Delphi 2007 projects (*.dproj)
- Microsoft Visual Studio 2005/2008/2010/2012 solutions (*.sln)

2. System Requirements

To install and use MSBuild Sidekick your system must meet the following requirements:

- Microsoft[®] Windows 7, Microsoft[®] Windows 8, Microsoft[®] Windows Server 2008, Microsoft[®] Windows Server 2012 operating system installed
- Microsoft[®] .Net 4.5 Framework installed
- 10 Mb of disk space for the program files

3. Installation Instructions

To install MSBuild Sidekick, double-click the MSBuild Sidekick MSI file and follow the installation screens instructions.

To uninstall MSBuild Sidekick,

- 1. Open the Windows "Control Panel"
- 2. Open the "Add/Remove Programs" from "Control Panel"
- 3. Select "MSBuild Sidekick v4.5" from the list of installed applications
- 4. Click the "Add/Remove" button to uninstall MSBuild Sidekick application and follow the uninstaller screens instructions.

4. Purchasing MSBuild Sidekick

If you have installed MSBuild Sidekick v4.5 without purchasing a license and registering your copy, the trial will expire in 15 days after the installation, and the application functionality will not be available thereafter. If you wish to purchase license, please visit our web site at <u>http://www.attrice.info/msbuild</u> and follow the purchasing and registration instructions on the site.

5. MSBuild Sidekick Features List

MSBuild Sidekick v4.5 provides the following features:

- View project structure in categorized project tree view
- View project structure in sequential project tree view (elements appear in the same order as in build project)
- Show or hide imported elements in project tree view as desired
- Add new or delete selected elements
- Edit selected element's properties and comments using properties grid
- Edit elements properties in a list (for container elements such as PropertyGroup etc.)
- Edit selected element's properties and comments using raw XML view
- Edit project file using raw XML view
- Add tasks to targets using categorized "Add task" dialog
- Use properties, items and item metadata Intellisense support when editing elements

- Change elements order in sequential project tree view or in elements list (for container elements)
- Perform "intelligent" rename of the elements throughout the project (all relevant elements are renamed)
- Delete elements throughout the project (no "dead" references left behind)
- Navigate between elements and element views (properties, list or raw XML) using backward/forward navigation
- Navigate between elements using "Find Element" dialog
- Navigate between elements using "Jump To" function in element properties
- Use integrated MSDN help (either from locally installed MSDN collection or from online MSDN)
- View targets and tasks execution graph
- Specify global or project-specific properties to be used in build session
- Modify environment properties per project for the build session
- Select targets for the build session
- Build project
- Stop building project. MSBuild API doesn't support stop build functionality by its own therefore MSBuild Sidekick terminates MSBuild process in order to stop build.
- Review build results using console log (similar to command-line MSBuild console log)
- Review build results using errors/warnings log
- Review build results using structured detailed log; navigate to specific elements from the log
- Debug project execution
- Pause debugging by setting breakpoints on Tasks/Targets
- Step-by-step debugging
- Analyze build process by viewing built and skipped Targets/Tasks on Debug Diagram
- View real-time Property and Item Include/Exclude values on Globals window while debugging
- View Property and Item Include/Exclude values changed on previous debug step
- View values of Properties and Items involved in Task execution on Autos window
- Edit MSBuild Extension Pack task property values with task-type specific editors
- View MSBuild Extension Pack tasks online help in Help window

MSBuild Sidekick v4.5 supports all elements in MSBuild 2.0, 3.5, 4.0 and 4.5 schema, namely:

- Project element (ToolsVersion attribute is supported only for MSBuild 3.5 projects)
- ProjectExtensions element
- Choose element
- Import element
- UsingTask element
- PropertyGroup element (can be nested under Target elements for MSBuild 3.5 projects)
- Property element
- ItemGroup element (can be nested under Target elements for MSBuild 3.5 projects)
- Item element (Remove attribute is supported only for MSBuild 3.5 projects)
- Target element
- Task element
- Tasks' Output element
- ItemDefinitionGroupElement (only for MSBuild 3.5 projects)

6. Graphic User Interface Outline

The application's user interface consists of main window menus (labeled 1 on the screenshot below), main window toolbars (labeled 2), status bar (labeled 12) and nine dockable windows: Project Tree window (labeled 3), Properties window (labeled 4), Elements window (labeled 5), Raw XML window (labeled 6), Help window (labeled 7), Debug window (labeled 8), Logs window (labeled 9), Globals window (labeled

10) and Autos window (labeled 11),. Please note that on the screenshot six windows are docked into tabbed view (windows 5, 6, 7, 8, 10, 11).

🗘 MSBuild Sidekick v2.3 - IssueTester.csproj				
<u>Eile Edit View Build Tools H</u> elp 1				
🗋 💕 🛃 🖏 - 🗙 🕆 🔱 🍢	🏣 🏗 🏔 🕜 🕥 🗉 🖃	🗈 🕨 🕾 💷 🎊 🔼		
Project Tree 3 4 × 4 P	Properties Elements Raw XML	Help Debug	• ×	
E S Project	<u>ۇ</u> ↓ <u>∢</u> <u>5</u> 6	7 8		
🕀 🐠 💓 PropertyGroup 🛛 🗖 Ge	eneral		_	
🕀 🐠 🎉 PropertyGroup 🛛 🛛 📔	ementType	Project	_	
ItemGroup Im	nported	False		
ItemGroup	oject Specific			
Emport (\$(MSBuildToi De	aultTargets	Build		
ElementType The type of MSBuild element				
Log 9			μ×	
Show Log: Edit Project Log 🔹 🐳	(11)			
Message	Details	Proj		
1 27 Failed to load assembly	Assembly cannot be found	or is not a valid .NET assembly C:\		
i) 28 Imported project has been load	led Application loads imported	project regardless the Conditio C:\		
(i) 29 Imported project has been load	led Application loads imported	project regardless the Conditio C:\	•	
🗐 Log 👦 Autos 🞘 Globals				
Build failed at 10 2.2009 11,:23 12				

6.1 Menus

The main application window menus provide access to the following functional areas.

1. File menu groups functions related to project files operations, namely:

File	Edit	View	Build
	New	Ctrl+	٠N
2	Open	Ctrl+	0
	Close		
H	Save	Ctrl+	-s
	Save As,		
	Reload		
	Recent Projects 🔹 🕨		
	Exit		

- New menu creates new empty project file
- **Open** menu opens existing project file
- Close menu closes currently loaded in the application project file
- Save menu saves changes made to currently loaded project file
- Save As menu allows saving current project file under a new name
- Reload menu reloads currently loaded project file from the disk drive

- **Recent Projects** menu provides shortcut to load previously opened project into the application; the list of nine last project paths is maintained
- **Exit** menu exits the application
- 2. Edit menu groups functions related to editing currently loaded project file:



- Add menu appends new element to project (user will be able to select new element type based on MSBuild schema version and element currently selected in the project tree)
- Remove menu removes selected element and all contained elements from project
- Move Up menu moves selected element up in elements order in project XML source
- Move Down menu moves selected element down order in project XML source
- Show Imported Elements menu shows or hides imported elements in the Project Tree window
- **Ordered Tree** menu switches Project Tree window to ordered mode (see below on the mode description)
- **Categorized Tree** menu switches Project Tree window to categorized mode (see below on the mode description)
- Find Element menu opens "Find element" dialog
- 3. View menu groups functions related to application look and feel:

Viev	<u>B</u> uild <u>T</u> ools <u>H</u> elp		
÷.	Project Tree		
	Log Window		
9 <u>=</u>	Debug Window		
A	Watch Window		
3	Autos Window		
G	Backward		
Ð	<u>F</u> orward		
Đ	Expand All Tree Nodes		
Ξ	Collapse All		
~	Standard Toolbar		
	<u>V</u> iew Toolbar		
~	Build <u>T</u> oolbar		

- Project Tree menu shows or hides Project Tree window
- Log window menu shows or hides Log window
- **Debug window** menu shows or hides Debug window

- Globals window menu shows or hides Globals window
- Autos window menu shows or hides Autos window
- **Backward** menu navigates back in project's navigation history (history includes selected element and element view window)
- **Forward** menu navigates forward in navigation history (history includes selected element and element view window)
- Expand All Tree Nodes menu expands all tree nodes in the Project Tree window
- Collapse All menu collapses all tree nodes in the Project Tree window
- Standard Toolbar menu shows or hides Standard toolbar in the main application window
- View Toolbar menu shows or hides View toolbar in the main application window
- r in the main application window

4. **Build** menu groups functions related to building projects:

<u>B</u> uild		<u>T</u> ools <u>H</u> elp			
🕨 Debug		ebug	F5		
≡⊳	Build Project Ctrl+F		Ctrl+F5		
	Stop Building Project		Stop Building Project Shift+F5		
٩ī	Debug Step		F11		
	Build Options				

- Debug menu starts debugging for the currently loaded project
- Build Project menu starts building for the currently loaded project
- Stop Building Project menu stops building/debugging
- Debug Step menu executes current Target/Task and highlights next Target/Task to be executed
- Build Options menu invokes Build Options dialog

5. Tools menu groups miscellaneous functions:

Tools		Help
C		pen Source in Visual Studio
	Open Source in Notepad	
Vie		ew Targets Diagram
Override Ta		verride Target
Qptions		ptions

- Open Source in Visual Studio menu opens the currently loaded project in Visual Studio if installed
- **Open Source in Notepad** menu opens the currently loaded project in Notepad
- View Targets Diagram menu invokes Targets Diagram dialog for the currently loaded project
- Override Target menu invokes Override Target dialog for the currently loaded project
- Options menu invokes application's Options dialog
- 6. Help menu groups functions related to help and support:

Help		
	Register	
	Check for Updates	
	Feedback	
	Home Page	
	About	

- Register menu invokes application registration dialog
- Check for Updates menu checks for application updates
- Feedback menu invokes default mail client with new email template for the feedback
- Home Page menu invokes application web page in user's web browser of choice
- About menu invokes About dialog

6.2 Toolbars

The application toolbars provide quick access to the functionality also available in the application menus.

Standard toolbar

🗋 🐸 📓 🌗 • 🗙 👔 🛊 🔛 🚵

- New button is identical to File → New menu
- **Open** button is identical to **File** \rightarrow **Open** menu
- Save button is identical to File -> Save menu
- Add New Element button is identical to Edit → Add menu
- **Remove Element** button is identical to **Edit**→**Remove** menu
- Move Element Up button is identical to Edit → Move Up menu
- Move Element Down button is identical to Edit → Move Down menu
- Show Imported Elements button is identical to Edit—Show Imported Elements menu
- Ordered Tree View button is identical to Edit -> Ordered Tree menu
- Categorized Tree View button is identical to Edit—Categorized Tree menu
- Find Element button is identical to Edit—Find Element menu

View toolbar

30100

- Navigate Backward button is identical to View -> Backward menu
- **Navigate Forward** button is identical to **View** \rightarrow **Forward** menu
- Expand All Tree Nodes button is identical to View-Expand All Tree Nodes menu
- Collapse All Tree Nodes button is identical to View-Collapse All Tree Nodes menu
- Show Log Window button is identical to View -> Log Window menu

Build toolbar

🗈 🕨 🚰 💷 🎯

- **Build** button is identical to **Build**→**Build Project** menu
- **Debug** button is identical to **Build** → **Debug**
- **Debug Step** button is identical to **Build** → **Debug Step**
- Stop Building Project button is identical to Build -> Stop Building Project
- Build Options button is identical to Build → Build Options menu

6.3 Project Tree Window

The Project Tree window displays elements in currently loaded project either as a hierarchical tree in order of appearance in underlying XML file or categorized by element type. By default, when project is loaded the

project tree is displayed in "Ordered" tree mode (as opposed to "Categorized" tree mode).



The mode can be switched using menu Edit \rightarrow Ordered Tree or Edit \rightarrow Categorized Tree (or corresponding buttons on Standard toolbar).

Every element displayed in the tree has an icon associated with it that denotes element type:

Element type	Icon
Project	<u>s</u>
Import	41
Using task	-
Property group	**
Property	<u>م</u>
Item group	
Item	E
Item definition group	
Item metadata	e
Target	±
Task	→
Output	+3
ProjectExtensions	
Choose	暴
When	ካ
Otherwise	101

In Categorized mode, all elements are grouped by the element type with category icons same as element type icons.

The user may control whether to display imported elements in the tree (in either mode) using $Edit \rightarrow Show$ Imported Elements menu (or corresponding button on Standard toolbar).

Clicking the left mouse button on any node in the project tree will select the node and will load into the Properties, Elements, Raw XML and Help windows the data corresponding to the element selected. Right-clicking the selected node will display the context menu containing the following menu items:



- Add (element) menu, where element types that may be added are dependent on the type of the element selected; the menu adds element to currently selected element node
- **Remove** menu removes currently selected element
- Move Up menu moves currently selected element up (within parent element)
- Move Down menu moves currently selected element down (within parent element)

Additionally, for Project and Import elements, there will appear **Open In New Window** menu, that will open the project file corresponding to the element selected in a new instance of MSBuild Sidekick.

6.4 Properties Window

Properties window displays attributes data specific to the element selected in Project Tree window.

	Properties Elements Raw XML Help 🗸 🗙			
	2 ↓ <i>a</i> , -			
Ξ	General			
	Condition		~	
	ElementType	Target		
	Imported	False		
	Name	Target1		
Ξ	Target Specific			
	DependsOnTargets			
	Inputs			
	Outputs			
Co If ar	Condition Condition to be evaluated. If the condition evaluates to false, the target will not execute the body of the target or any targets set in the DependsOnTargets attribute			
S Comments				
			*	

For every element the following properties are displayed:

- Condition defined on the element
- Element type
- Imported indication (whether element is imported or local to the project)
- Element name

The rest of the properties are specific for the element type; if the element is local to the project (not imported) those properties are editable.

In the textbox located under the properties grid the element's comments are displayed; the box can be shown/hid using the s button. Similar to the element properties, for the local (not imported) elements the comments are editable.

On the window toolbar are displayed additional functional buttons.

Properties Elements Raw XML Help

First two buttons (\bigcirc and \bigcirc) control the properties display mode – by category or by alphabet correspondingly.

For the properties window for elements of Property elements, there are additional Condition Edit buttons in

the toolbar ($\frac{1}{2}$ $\frac{1}{2}$). Using those buttons, the user may quickly specify "property is empty" or "property file exists".

For elements that contain references to the other items or properties, Jump To button (

6.5 Elements Window

Elements window displays elements contained in the selected element in tabular format. The tab is relevant for the following element type: PropertyGroup elements, ItemGroup elements, Item elements, Target elements, Task elements and ItemDefinitionGroup elements (for MSBuild 3.5 projects).

Properties Elements Raw XML Help						
🗊 🗙 🕋 🗈 🌲						
Item Name	Include	Exclude	Remove	Condition		
Item	None					
Item	None1					

The columns in the table differ depending on the type of the contained element; the data in the columns are editable for local (not imported) elements.

The window toolbar buttons provide the following additional functions:

- Add Element (¹) button opens element adds new element to the table
- Delete button (\times)removes selected element(s)
- Properties button ()) selects the element in the table in the project tree and switches to properties window view for that element
- Move up button (1) moves selected element up in list and xml
- Move down button (\checkmark) moves selected element down in list and xml

6.6 Raw XML Window

Raw XML window displays raw XML source for the element selected in the project tree. The XML can be edited for local (not imported) elements.



The XML editor provides color coding for XML elements and attributes and supports auto-completion for properties, items and item metadata elements.

The window toolbar buttons provide the following additional functions

- Save button () commits changes performed in raw XML. If changes cannot be saved due to the error, the corresponding message appears in Edit Project Log window
- Undo button (*) reverts to the original XML contents, undoing the changes made by user
- Word Wrap button () switches word wrap mode to on and off
- Auto-format button (\blacksquare) formats XML in the editor with automatic indentation
- Find In XML button (th) Find In XML dialog to search the XML for specified text

6.7 Help Window

Help window displays MSDN article for the selected MSBuild element type; for Microsoft tasks task-specific help will be displayed.



Depending on the application settings (set through **Tools** \rightarrow **Options** menu, **Application Configuration** tab, MSDN Help Source property), either local MSDN collection is used or a web version. By default, the application uses the web version.

6.8 Targets Diagram Window

Targets Diagram window displays targets and tasks execution order without actually execution a build. Note that MSBuild Sidekick doesn't evaluate Condition attribute values of various elements whild building Targets Diagram, so actual execution order can only be found on Diagram Window during (or after) build execution.

Targets Diagram window toolbar contains the following controls:

- Zoom In (^(a)) button allowing user to zoom in diagram image
- Zoom Out () button allowing user to zoom out diagram image
- Save as Image (📕) button that saves diagram as image file on a file system
- **Print** () button that prints diagram on a printing device
- Show/Hide Tasks (📫) button allowing to show/hide Tasks defined under Targets

Targets Diagram						
ର୍ ର୍ 🖵 🛃 🛃	Q Q 🖬 🚭 🔸					
Filter All Targets	Reset to Defaults					
Targets to build:	Initial Targets	CheckSettingsForEndToEndIteration				
Name	Dependencies Targets	BeforeCompile)				
AfterClean						
AfterCompile		CoreCompile				
AfterDropBuild						
AfterEndToEndIteration						
AfterGet						
AfterLabel		(At Compile)				
AfterOnBuildBreak						
AfterTest						
📃 BeforeClean		Before l'est				
BeforeCompile						
🔲 BeforeDropBuild		CoreTest > #MakeDir > #MSBL				
BeforeEndToEndIteration		V				
E BeforeGet		4 AfterTest				
BeforeLabel		✓				
🔲 BeforeOnBuildBreak						
		Close				

6.9 Debug Window

Debug window displays build diagram with Targets/Tasks defined in project. Debug window allows one to set/remove breakpoints on Targets/Tasks while analyzing build sequence and viewing the diagram. During execution of the build, MSBuild Sidekick colors diagram elements according to their status in build (Built or Skipped).

Debug window toolbar contains the following controls:

- Zoom In (^(C)) button allowing user to zoom in diagram image
- Zoom Out () button allowing user to zoom out diagram image
- Save as Image (📕) button that saves diagram as image file on file system
- **Print** () button that prints diagram on printing device
- Show/Hide Tasks (🔹) button allowing to show/hide Tasks defined under Targets

Double click on diagram element jumps to specific Target/Task element in Project Tree window. Using mouse right click context menu new breakpoint can be set or all breakpoints can be removed. Note that MSBuild Sidekick may not stop on a breakpoint if MSBuild engine skips the Target/Task during the build.

To show/hide Task elements "Show/Hide Tasks" button (📩) can be used. Note that debugger won't stop on breakpoint set on Task if the task is hidden.

In the panel located under Debug diagram all possible diagram icons and status colors are displayed; the panel can be shown/hidden using the 🗵 button.



6.10 Logs Window

Logs window displays information related to project editing and build events (such as errors and warning occurring during editing etc.)

Logs window toolbar contains the following controls:

- Show Log combo box allowing user to choose the visible log (logs available are Edit Project log, Build Console log, Build Errors/Warnings log and Build Detailed log)
- Clear Log button that clears the current log
- Pause Log button pauses (or resumes, once paused) the logging output during the build

Edit Project log displays editing events starting with project load; the events are either notifications of success

Log Window	, — — — — — — — — — — — — — — — — — — —			η×
Show Log:	Edit Project Log 💿 🔹 🛒 🕕			
	Message	Details	Project	
(i) 1 F	Project has been loaded successfully	Well-formed MSBuild project.	C:\Temp\new1.proj	

The log data is organized in a table with the following five columns:

- Event type icon, related to message type (information/warning/critical)
- Event message order number
- Event message
- Event message detailed text
- Filename of the project in which event has occurred

Build Console log contains build-related messages and is populated during the build.

Log Window	4 ×
Show Log: Build Console Log 🔹 🔫 🕕	
Build succeeded. O Warning(s)	^
0 Error(s)	=
Time Elapsed 00:00:00.35	~

The log contents are identical to MSBuild command-line log (and adhere to the same color schema with errors displayed in red etc.). The verbosity of the logging is controlled through **Tools** \rightarrow **Options** menu, **Loggers** tab, **Console Logger** Verbosity property (by default Verbosity is set to Normal)

Build Errors/Warnings log contains error and warning messages related to the last build and is populated during the build.

Log Window				X
Show Log: Bu	uild Errors/Warn	nings 🔽 式 🕕		
Event Type	Sender	File	Message	^
Error	MSBuild	C:\Temp\test.sln.proj	The project file "Common\Common9.csproj" was not fo	
🕺 Error	MSBuild	C:\Temp\test.sln.proj	The project file "Controllers\Controllers9.csproj" was n	=
🕺 Error	MSBuild	C:\Temp\test.sln.proj	The project file "Controls\Controls9.csproj" was not fo	_
🕺 Error	MSBuild	C:\Temp\test.sln.proj	The project file "IssueTester\IssueTester9.csproj" was	
🕺 Error	MSBuild	C:\Temp\test.sln.proj	The project file "SidekicksApp\TeamFoundationSidekick	
Rrror	MSBuild	C:\Temn\test.sln.nroi	The project file "SidekicksPackage' TeamFoundationSid	×

The log data is organized into a table with six columns:

• Event Type icon (either error or warning icon)

- Sender (either MSBuild engine or specific MSBuild task)
- Path to the file in which error/warning occurred
- Error or Warning message

Build Detailed log is populated during the build and provides hierarchical tree view of the targets and tasks executed in the build.

Log Window							ųΧ
Show Log:	Build Detailed Log	- 🛼 🕕) Status: All		•		
Element			Time	File		Diagnostic	
😑 🗿 Proj	ect	Image: A start of the start	00:00:00.31	2 new1.proj			
🗉 🗄 1	arget1		00:00:00.25	0 new1.proj			

The data in the log are organized into a table with five columns:

- Element type column, containing element type name and the corresponding icon
- Build status icon for the element with the following values
 - \circ Successfully executed (
 - \circ Failed (
 - Skipped in build (*); usually occurs due to condition evaluated to false or if the inputs are upto-date
- Element execution time
- Project file the element belongs to
- Diagnostic message, containing additional information provided by MSBuild Engine

Additional **Status** combo box is located in toolbar, and may be used to filter the elements appearing in the log as following:

- All is equivalent to no filter showing all elements in log
- Succeeded filter shows all successfully executed targets and tasks
- Failed filter shows all failed targets and tasks
- Skipped Targets filter shows only skipped targets

Skipped Tasks filter shows only skipped tasks (and their parent targets)

6.11 Globals window

Globals window displays values for Properties or Items defined in project as they are evaluated during the project debug session.

Globals window toolbar contains the following controls:

- Show combo box allows filtering currently visible elements (Properties or Items)
- **Filter** text box allows filtering currently visible elements based on element name (only elements with the name that starts with the text specified are displayed)

Globals		×
Show: Properties - Filter:		
Name	Value	•
TargetFrameworkDirectory	C:\WINDOWS\Microsoft.NET\Framework\v3.5;C:\WIN	
TargetFrameworkSDKDirectory	C:\Program Files\Microsoft SDKs\Windows\v6.0A\	
_AfterCompileWinFXInternalDependsOn	PrepareResourcesForSatelliteAssemblies Mer	
_CompileTargetNameForLocalType	_CompileTemporaryAssembly	
_DebugSymbolsProduced	true	
_DeploymentApplicationManifestIdentity	ConsoleApplication5.exe	
_DeploymentBuiltUpdateInterval	0	-1

The Globals window data is organized as a table with the following columns:

- Property/Item name (depending on Show combo box selection)
- Property value (if "Properties" is selected in Show combo box)
- Include and Exclude values (if "Items" is selected in Show combo box)

When stepping through the project execution in debug mode, Properties/Items that have been changed on previous step are highlighted in red.

6.12 Autos window

Autos window is similar to Globals window but displays only Properties/Items that are defined as input or output parameters for Target/Task element to be executed at the next step, when stepping through the project execution.

Autos		×
Show: Properties - Filter:		
Name	Value	•
_FindDependencies		
AllowedReferenceAssemblyFileExtensions	s DD .exe;100 .d100	
AllowedReferenceRelatedFileExtensions	DU .pdbjDU .xmDD	
AssemblySearchPaths	00 {CandidateAssemblyFiles}00 00 {HintPathFro	
AutoUnifyAssemblyReferences	false	
BuildingProject	true	
IgnoreInstalledAssemblyTables		•

7. How-To Notes

7.1 How To Use Tree Modes Effectively

The project tree window (see *Project Tree Window* section above) displays elements in project either in the order of appearance in the project or grouped by category.

Ordered mode is especially useful when changing a project; new elements are added by default to the end of the file or to the end of currently selected element children elements (for new properties, items and item metadata elements). The order of the element can be changed by using Move Up/Move Down menus/toolbuttons.

But while it provides full control over order of elements in the file, categorized mode is more convenient when viewing build file contents. For example, it provides alphabetically sorted list of all properties, and together with ability to toggle the visibility of imported ones, it becomes very easy to find specific property or view the list of all existing ones.

7.2 How To Add Tasks To Target

To add task to a target, you will usually use Add Task dialog. The dialog displays alphabetically sorted list of tasks for several categories. The task categories available are:

- All Tasks list displays all tasks available in different categories combined
- MSBuild Tasks list displays tasks from Microsoft.Build.Tasks assembly (MSBuild standard tasks)
- Team Build Tasks list displays tasks from Microsoft Team Build tasks assembly
- Imported Tasks list displays tasks used throughout imported projects (for currently loaded project)
- Local Tasks list displays used in currently loaded project
- *Custom Tasks* list displays tasks from Custom Using tasks assemblies (as configured in Tools→Options menu; see *Advanced topics* for details)

A	dd Task		×	
	ALLA	Tasks	^	
	All Tasks	AL		
	All T Goldo	AppDomainIsolatedTaskExtension		
		AspNetCompiler		
	<u>%</u>	AssignCulture		
		AssignProjectConfiguration		
	MSBuild	AssignTargetPath		
	Tasks	Call arget		
	iii+	CombinePath		
	-	Convert I oAbsolutePath		
	Team Build			
	Tasks	LreateLSharpManifestResourceName		
	-	Createmanifestriesourceiname		
	Imported			
	Tasks	Create Lemporary VLProject		
	1 Gaila			
		Delete		
		Error		
	Local Tasks	Even		
		Find InderPath		
	0.4	Formatilit		
		FormatVersion		
	Custom	GenerateApplicationManifest		
	Tasks	GenerateBootstranper		
		GenerateDeploymentManifest	~	

When task is selected in the list, double-clicking the task selected or hitting Add button will add the task to the target. You can add multiple tasks before closing the dialog using Close button (or hitting Escape).

There are several ways to add a new task to the target:

- 1) Using Add Element button (⁵¹) on Target element
 - Select specific target in the project tree
 - Click on Standard toolbar Add Element button and choose Task from dropdown list
 - In the displayed Add Task dialog select a task. New task will be added to the end of target's task list
- 2) Using Add Element button (¹) on Task element
 - Select specific task in project tree
 - Click on Standard toolbar Add Element button and choose Task from dropdown list
 - In the displayed Add Task dialog select a task. New task will be added before the task selected in the project tree
- 3) Using Elements window
 - Select specific target in the project tree
 - Switch to Elements window
 - Click Add Element (¹) button on window's toolbar
 - In the displayed Add Task dialog select a task. New task will be added to the end of target's task list
- 4) Using Raw Xml
 - Select target in project tree
 - Switch to Raw Xml window
 - Edit xml to add new task
 - Save changes by pressing 😼 button

7.3 How To Specify Tasks Output Parameters

When you add a task that has output parameters, only the mandatory output parameters will be added by default. To add optional output parameters, select the task in the project tree view, click the right mouse button, select **Add Output** menu and then specific parameter from the list, as shown on the picture below:



Once the output parameter is added, you will have to specify either property or item to use in output. **Note**: There is no need to specify \$ or @ qualifiers when specifying item or property names in output parameters.

7.4 How To Add New Elements

There are several ways to add new elements; depending on the element type and the number of the elements to add some of them are more efficient than the others.

Elements of any type may be added using Standard toolbar Add Element button (1). Clicking on the button arrow will display list of elements that may be added (depending on currently selected element).

To add many properties or items (to property group or item group element correspondingly), one may use Elements window. The table layout of the elements window and the ability to switch between the columns using Tab key allows for fast and convenient addition or modification of similar elements.

And finally, the elements may be added in raw XML window (see below).

7.5 How To Use Raw XML

If you are comfortable with XML syntax for MSBuild, you may want to use raw XML for certain elements editing (such as adding a lot of similar properties to a property group or lots of similar items to an item group).

While editing raw XML it is helpful to keep in mind several things:

- One can always revert to the original XML by using Undo button on window's toolbar
- The changes to XML are committed only after Save button on window's toolbar is pressed
- If the changed XML is not a valid XML conforming to MSBuild schema, the changes will *not* be committed upon Save; the user is supposed to review Project Edit log and fix the problems for the next Save to succeed
- Rename and delete refactoring support (see below) is active for raw XML editing as well; as XML editing may result in names changed/element deletion for several elements at once, the refactoring dialogs will show data for all such elements in one dialog

7.6 How To Use Conditions Edit Shortcuts

Frequently, the condition set on a property is to check for an empty value or for the existence of the file/folder, specified by the property.

In the Properties window toolbar there are Condition Edit Shortcut buttons that provide convenient shortcuts for specifying such conditions.

For the selected property (e.g. *Property1*), clicking Check For Empty Value button (\pounds) will add the following condition: '\$(*Property1*)' == ''

Clicking Check If File Exists button (^k) will add the following condition: *Exists('\$(Property1)')* **Note:** when a condition is added using Condition Edit Shortcut buttons, the previous condition value will be erased.

7.7 How To Navigate Around

Application saves navigation history (project tree view mode, tree node selected in the project tree, window for the tree node).

To navigate the history use the following controls:

- *View* \rightarrow *Forward* menu or Standard toolbar Navigate Forward button (\bigcirc)to navigate forward
- *View* \rightarrow *Backward* menu or Standard toolbar Navigate Backward button (\bigcirc) to navigate backward.

Additionally, it is possible to navigate to Properties, Items or ItemMetadata elements referenced in the selected element attributes. The Jump To Reference button (\triangleq) is located in the Properties window toolbar:

	Properties Elements Raw XM	IL Help - X
	2↓ 2	
Ξ	Gener 🔗 \$(RootPath)	
	Condition	
	ElementType	Item
	Imported	False
	Name	Item
Ξ	Item Specific	
	Exclude	
	Include	\$(RootPath)*.txt
	Remove	

Clicking the button will display the list of referenced elements; selecting the element from list will select the element in the project tree.

Note: if the project contains several elements with same name, Jump To resolution dialog will appear.

Jump To					
Note There are several elements with name "IntermediateOutputPath". All of them displayed in list below. Clicking "OK" button will perform jump to selected element in project tree.					
Element		Project			
🍳 IntermediateOutputF	Path	Microsoft.Common.targets			
PartermediateOutputF	Path	Microsoft.Common.targets			
P					
1 <inter< td=""><td>rmediateOutputPath</td><td>Condition="</td><td></td><td></td></inter<>	rmediateOutputPath	Condition="			
'\$(P)	latformName)' != 'A	nyCPU' ">\$ (
Baselı	ntermediateOutputPə	th)\$(PlatformName))\\$(Configur	ation)\	
Inte</td <td>ermediateOutputPath</td> <td>1></td> <td></td> <td></td>	ermediateOutputPath	1>			
				100	
				<u>×</u>	
			ОК	Cancel	

7.8 How To Rename Elements

When an element such as property, item, item metadata or target is renamed, the application performs dependency check and displays all references found in "Rename Element" dialog.

Rename Element "Configuration"						
Results You are going to rename element "Configuration". Found elements contain references to this element. Clicking "Rename and Update" button will rename element and update its references. Clicking "Rename" button will rename element and keep references with previous element's name.						
Element	Туре	Attribute Name	Attribute Value			
Configuration	Property	Condition	'\$(Configuration)' == "			
PropertyGroup PropertyGroup	PropertyGroup PropertyGroup	Condition Condition	'\$(Configuration) \$(Platform)' == 'Debug A '\$(Configuration) \$(Platform)' == 'Release			
P						
<pre>1 <configuration condition=" '\$ (Configuration) ' == '' ">Debug</configuration></pre>						
Jump To		Rena	ame and Update Rename Cancel			
Total elements found : 3			.::			

Clicking "Rename and Update" button will rename element and update all its references with a new name. Clicking "Rename" button will rename element and but not the references.

Clicking "Jump To" button will select the reference element in the project tree.

7.9 How To Delete Elements

When an element such as property, item, item metadata or target is deleted, the application performs dependency check and displays all references found in "Delete Element" dialog.

Delete Element "Testi	Property"						
Results You are going to delete element "TestProperty". Found elements contain references to this element. Clicking "Delete" button will delete element, so references will probably become invalid.							
Element	Туре	Attribute Name	Attribute Value				
CallTarget	Task	Targets	\$(TestProperty)				
2							
1 <calltarget targets="\$(TestProperty)"></calltarget>							
				~			
Jump To				Delete Cancel			
Total elements found : 1				.::			

Clicking "Delete" button will delete element, and invalidate the references. Clicking "Jump To" button will select the reference element in the project tree.

7.10 How To Use Build Options

Setting project properties

To set properties for the project build (similar to command-line MSBuild /p switch) **Build** \rightarrow **Build Options**, **Project Properties** tab or **Tools** \rightarrow **Options**, **Global Properties** tab may be used. The difference between those properties lists is that Project Properties will apply only to currently loaded project, whereas Global Properties will apply to all projects.

To set property related to currently loaded project:

- Click on **Build** → **Build Options** menu or Build toolbar **Build Options** button
- Click on Project Properties icon and open Project Properties tab

Bu	uild Options		×
		<u>⁰ X</u>	
	Engine	Name	Value
	Flopenies	Configuration	Release
		1	
	Enviroment Properties		
	Project Properties		
	<u>*</u>		
	Targets		
	Show Option	s dialog before build	Build Cancel

Page contains table of properties names and values, with property name in Name column and property value in Value column. Putting values in new table row will create new property

To delete property, use Remove Property button on Project Properties tab toolbar.

Setting environment properties

Environment properties values can be redefined using **Build**→**Build Options** menu, Environment Properties tab.

Build Options			×
	Reset to Defaults		
Engine	Name	Value	^
Properties	ALLUSERSPROFILE	C:\Documents and Settings\	
	APPDATA	C:\Documents and Settings\	
Enviroment	CommonProgramFiles	C:\Program Files\Common Files	
Properties	COMPUTERNAME	CUREDONE	
P 1	ComSpec	C:\WINDOWS\system32\cm	
Project Properties	FP_NO_HOST_CHECK	NO	
	HOMEDRIVE	C:	1
	HOMEPATH	\Documents and Settings\eu	1
	lib	C:\Program Files\SQLXML 4	1
	LOGONSERVER	\\CUREDONE	1
	MSBuildExtensionsPath	C:\Program Files\MSBuild	1
	MSBuildExtensionsPath32	C:\Program Files\MSBuild	1
	NUMBER_OF_PROCESSORS	1	
	ns	Windows NT	×
Show Options	dialog before build	OK Cancel	

To change the property value, select property from table and type new property value in Value column For the redefined property the name will be colored in red.

To reset environment properties to default, click **Reset to Defaults** button on Environment Properties tab.

Setting build targets

By default application will build targets defined by project's InitialTargets and DefaultTargets attributes values.

Build Options		X
Engine Properties	Filter All Targets	Reset to Defaults
	Name	Dependencies Targets
Properties	🗹 Build (default)	\$(BuildDependsOn)
	CheckForCompileOutputs	
	CheckForInvalidConfigurationA	
Project	CleanGetCurrentAndPriorFileWr	_CheckForCompileOutputs;_SG
riopenies	CleanRecordFileWrites	_CleanGetCurrentAndPriorFileWr
<u></u>	ComputeNonExistentFileProperty	
Targets	CopyAppConfigFile	
	CopyFilesMarkedCopyLocal	
	CopyFilesToPublishFolder	
	CopyManifestFiles	PrepareForBuild
, V Show Option	ns dialog before build	Build Cancel

To set build targets different from the defaults, select the targets to build from the list. The list can be filtered to show all/local/imported targets using Filter combo box. To change the order of targets in build, use Move Up/Move Down buttons.

Note: Targets defined by Project's InitialTargets attribute value will be built always regardless of the list selections.

To reset the target selection to defaults, use Reset to Defaults button.

7.11 How To Use Target Diagram

To review the targets (and tasks) execution order without actually execution a build, one may use Target Diagram dialog; the dialog can be invoked using **Tools** \rightarrow **Target Diagram** menu.

Targets Diagram		×
ର୍ ର୍ 🖵 🛃 🔒		
Filter All Targets	💌 🛛 Reset to Defaults 👔 🌗	
Targets to build:	Initial Targets	
Nesse		
Name	Dependencies Largets	BeforeCompile
AfterClean		
AfterCompile		CoreCompile → MakeDin → 44C
AfterDropBuild		
AfterEndToEndIterati	ion	After Compile
AfterGet		
AfterLabel		
🔲 AfterOnBuildBreak		
AfterTest		
🔲 BeforeClean		BeforeTest
BeforeCompile		↓ ↓ ↓
BeforeDropBuild		CoreTest > #MakeDir > #MSBL
🔲 BeforeEndToEndItera	ation	
🔲 BeforeGet		AfterTest
🔲 BeforeLabel		
🔲 BeforeOnBuildBreak		
		Close

Once the dialog opens, you may want to specify the targets to review (i.e. the targets initially invoked in the build). If no targets are specified by user, project default targets will be used.

Note: also by default project's initial targets are always displayed; to hide them uncheck Initial Targets checkbox.

Placing the mouse over each element on the diagram will display tooltip window with element details; selecting the element on the diagram and clicking right mouse button will invoke the context menu with ethe following options;

- Jump To menu allows "jumping" to the element in project tree
- Information menu shows elements details as tooltip

When selected, the element on the diagram can also be moved to provide for nicer visual picture.

The diagram on the left will display the targets execution order starting with initial targets. By default, the tasks in every target will also be displayed; their display can be turned off using Show/Hide Tasks button on the window toolbar.

Also the toolbar contains the following functions:

- "Zoom In/Zoom Out" buttons may be used to change the zoom ratio of the diagram image
- "Save As Image" button may be used to save the currently displayed diagram into image
- "Print" button may be used to print the currently displayed diagram

The diagram is synchronized with project. If you add new targets to the project, they will be added to targets list automatically. If you add new tasks they will be displayed on the diagram (if corresponding targets are present on the diagram and Show/Hide Tasks button toggled).

7.12 How To Debug Step-by-Step

Build execution can be started in Debug mode. In this mode user may inspect Property and Item values defined in project as well as analyze build sequence. One may start debug in step-by-step mode by using Debug Step function (available as menu item and button) or start debugging from specific location in project by setting breakpoints at certain Targets or Tasks. To monitor build execution order open Debug window by clicking on **View** \rightarrow **Debug Window**.



Note that Debug window appears in floating mode when it is opened for the first time. To make Debug window docked drag-n-drop it at desired location onMSBuild Sidekick main application window.

Using Debug Step menu/button/hotkey

Make sure MSBuild Sidekick is in edit mode (project is not being built or debugged). There are three ways to start debug step-by-step mode:

- Click on **Build** → **Debug Step** menu
- Click on Debug Step (뛸) button at Build toolbar
- Press F11 key on keyboard

Once you perform one of the actions above, MSBuild Sidekick will start execute project in debug mode and initial Target/Task will be highlighted with bold border in Debug window.



Using Breakpoints

In project edit mode open Debug window, right click on needed Target/Task to display context menu, click on **Toggle Breakpoint** menu item.



Red circle will appear near the Target/Task name indicating that breakpoint is now set. After specifying desired breakpoints, there are three ways to start debugging:

- Click on **Build** → **Debug** menu
- Click on Debug () button at Build toolbar
- Press F5 key on keyboard

Once project execution reaches Target/Task with breakpoint set, MSBuild Sidekick will switch into step-bystep debug mode. Current Target/Task will be highlighted with bold border in Debug window.



7.13 How To See Property Values When Debugging

In order to see Property values or Item Include/Exclude attribute values start debugging project in step-bystep mode. When Target/Task is selected with bold border in Debug window click on **View** \rightarrow **Globals Window** and **View** \rightarrow **Autos Window**.



Globals and Autos windows will appear at bottom pane of main application window, displaying Properties and their values by default. To view Item Include/Exclude attribute values select Items in Show combo-box.

Globals	д	×
Show: Properties - Filter:		
Name Properties	Value	•
_AfterCompliewin=xinternalDependsOn	PrepareResourcesForSatelliteAssemblies; MergeLocaliz	
_CompileTargetNameForLocalType	_CompileTemporaryAssembly	
_DebugSymbolsProduced	true	
_DeploymentApplicationManifestIdentity	\$(AssemblyName).exe	
_DeploymentBuiltUpdateInterval	0	
_DeploymentBuiltUpdateIntervalUnits	Days	
_DeploymentDeployManifestIdentity	\$(AssemblyName).application	•

Note that Properties/Items that have been changed on previous debug step are highlighted in bold.

7.14 How To Use MSBuild Extension Pack Tasks

To use MSBuild Extension Pack (MEP) tasks you need the task assemblies need to be installed. The latest version of MEP tasks can be downloaded from <u>http://www.codeplex.com/MSBuildExtensionPack</u>.

MSBuild Sidekick detects the MEP task assemblies at every start. MEP task assemblies are supposed to be located at the following folders: *[MSBuild installation folder]\ExtensionPack\4.0* or

[MSBuild installation folder]ExtensionPack

By default [MSBuild installation folder] is C:\Program Files\MSBuild.

Once the tasks are installed, follow the steps below:

1. To use specific task in your project, add Task element and select

MSBuild.ExtensionPack.Framework.DateAndTime task on Custom Tasks page.

A	dd task		[x
	ALLA	Tasks]
		MSBuild.ExtensionPack.BaseTask		1
	All Lasks	MSBuild.ExtensionPack.CodeQuality.FxCop		
		MSBuild.ExtensionPack.Communication.Email		
	<u> </u>	MSBuild.ExtensionPack.Compression.Cab		
	- 🏠	MSBuild.ExtensionPack.Computer.ComponentServices		
	MSBuild	MSBuild.ExtensionPack.Computer.EnvironmentVariable		
	Tasks	MSBuild.ExtensionPack.Computer.EventLog		
		MSBuild.ExtensionPack.Computer.EventSource		
	MA .	MSBuild.ExtensionPack.Computer.Network		
		MSBuild.ExtensionPack.Computer.PerformanceCounters		
	I eam Build	MSBuild.ExtensionPack.Computer.Registry		
	Lasks	MSBuild.ExtensionPack.Computer.SystemDrive		
	2 🔺	MSBuild.ExtensionPack.Computer.WindowsService		
		MSBuild.ExtensionPack.FileSystem.Detokenise		
	Imported	MSBuild.ExtensionPack.FileSystem.File		1
	Tasks	MSBuild.ExtensionPack.FileSystem.Folder		
		MSBuild.ExtensionPack.FileSystem.Share		
		MSBuild.ExtensionPack.Framework.Assembly		
		MSBuild.ExtensionPack.Framework.AssemblyInfo		
	Local Lasks	MSBuild.ExtensionPack.Framework.DateAndTime		
		MSBuild.ExtensionPack.Framework.Gac		
	0.46	MSBuild.ExtensionPack.Framework.Guid		
	→ ☆	MSBuild.ExtensionPack.Framework.MSBuildHelper		
	Custom	MSBuild.ExtensionPack.Framework.Signing		
	Tasks	MSBuild.ExtensionPack.Framework.TextString		
		MSBuild.ExtensionPack.Framework.Thread	-	1
		,		-
				1
		Add Close	•	

3. To control the action of selected task, select relevant TaskAction in combo-box.

	Properties Elements Raw XML Help Debug 🗸 🗙					
	Name	MSBuild.ExtensionPack.Framework				
	Input Parameters					
	*Format	MM dd yyyy				
	Authority					
	LogExceptionStack					
	MachineName					
	Result					
	SuppressTaskMessages					
	TaskAction	Get 🗾				
	UserName	GetElapsed				
	UserPassword	Get				
	Output Parameters					
Ð	Result	_				
TaskAction						
8	S Comments					

4. Once TaskAction is selected, the task parameters list will be filtered automatically, so only task parameters relevant for the selected TaskAction will be displayed. Required input parameters will be marked with asterisk (see Format parameter on the screenshot above).

5. To view online help for the selected Task navigate to Help window.

Properties Elements Raw XML Help Debug - X		
🗏 Collapse All		
MSBuild Extension Pack Help (v3.5.1.0)		
DateAndTime Class		
Members Example See Also Send Feedback		
Valid TaskActions are:		
Get (Required: Format Output: Result)		
<i>GetElapsed</i> (Required: Format, Start Optional: End Output: Result)		
Remote Execution Support: NA		
Namespace: <u>MSBuild.ExtensionPack.Framework</u> Assembly: MSBuild.ExtensionPack (in MSBuild.ExtensionPack.dll) Version: 3.5.0.0 (3.5.1.0)		
🗏 Examples		
ХМЦ 🗈 Сору		
<pre><project <="" pre="" toolsversion="3.5"></project></pre>		

8. Configuration

8.1 Application Configuration

The configuration of the application can be changed through Options dialog, which is invoked using **Tools** \rightarrow **Options** menu.

Application configuration files and application logs are stored in "%APPDATA%/Attrice Corporation/BuildSidekick/2.0" folder.

8.2 Project Configuration

The project-specific configuration (the configuration for currently loaded project) can be changed through Build Options dialog, which is invoked using **Build** \rightarrow **Build Options** menu.

The options for the project are stored in the file <ProjectFileName>.<username> in the same folder where currently loaded project file is located.

9. Advanced topics

9.1 Solution files

Visual Studio solution files are not in MSBuild format. Therefore, to build them MSBuild engine converts the solution file into file in MSBuild format.

When a solution file is opened using MSBuild Sidekick, it is converted to MSBuild format and saved locally in the same folder as the solution file (file <solution name>.sln.cache is generated). As a part of solution conversion process MSBuild Sidekick can also create project metadata files for certain projects (files <project name>.metaproj are generated in project folders).

It is important to understand that when you modify the solution file project in MSBuild Sidekick, you are actually modifying completely separate project generated from the solution file, so any changes made will not propagate to the original solution file.

9.2 Override Target

In order to override targets from imported projects in local project, one may use Override Target dialog. The dialog is invoked using **Tools** \rightarrow **Override Target** menu.

Dialog contains list of targets from imported projects, which are not present in currently loaded local project. Add button creates target with the same name as the selected one in he currently loaded project.

9.2 Custom imports

Projects frequently used as imports can be added to the list of Custom Imports using **Tools** \rightarrow **Options** menu, **Custom Imports** tab. The project files listed in custom import pre-populate the imports combo box in the Add Import dialog; thus the user will be able to choose an import from the list rather than using Browse button and looking for the project.

9.3 Custom using tasks

Assemblies containing MSBuild tasks that are frequently used in projects can be added to the list of Custom Using Tasks using **Tools** \rightarrow **Options** menu, **Custom Using Tasks** tab. The tasks contained in the assemblies specified in Custom Using Tasks list will appear in the Custom Tasks category in Add Task dialog; when task is selected and added, UsingTask element with the reference to the assembly as specified in Custom Using Tasks will be added to the project.

9.4 Loggers

Custom MSBuild loggers can be configured to be used in the application builds using **Tools** \rightarrow **Options** menu, **Loggers** tab. The loggers thus setup will be hooked up the MSBuild engine instance during every build performed using MSBuild Sidekick.

10. MSBuild 2.0, 3.5, 4.0, 4.5 support

10.1 .NET Framework Version

You should have .NET Framework 4.5 installed in order to use MSBuild Sidekick. If you have previous MSBuild version please install MSBuild Sidekick 2.0 (.NET 2.0 or 3.5) or MSBuild Sidekick 3.0 (.NET 4.0).

10.2 MSBuild Schema Version Configuration

MSBuild Sidekick supports editing MSBuild 2.0, 3.5, 4.0 and 4.5 projects; only pre-requisite you need installed is .Net Framework 4.5. However, to build MSBuild 4.5 projects you need to have .Net Framework 4.5 installed.

When creating new projects or loading the projects into the application for the first time, the MSBuild version for the project being loaded is defined by the Initial Schema Version setting. The setting can be changed using **Tools** \rightarrow **Options** menu, on **Application Configuration** tab (by default, Initial Schema Version is set to 2.0).

A	pplication Opti	ons 🔀
	Application	🗆 Automatic Update
	Configura	Check for Updates Automatically True
		Period (days) 30
	<u> </u>	Connection
	Custom	Proxy Host
	Imports	Proxy Port 0
		Use Proxy False
	P	Dependencies Search
	Custom	Automatic Dependencies Search True
	Using Lasks	Dependencies Search for Raw XML True
	<u>∽•</u>	🗆 General
	Custom Task Editors	.NET Framework Version .NET Framework 2.0
		Env Properties in Auto-Completion True
		Initial Schema Version .NET Framework 2.0
		TFS Assemblies Path C:\Program Files\Microsoft Visual S
		🗆 MSDN Help
	GIODAI Properties	Local MSDN Collection MS.MSDNQTR.v90.en
	riopenies	Source Local
	Loggers	General
		OK Cancel

To change MSBuild schema version in the loaded project, use **Build** \rightarrow **Build Options** menu, **Engine Properties** tab, Schema Version setting.

B	uild Options		
	Engine Properties Enviroment Properties Project Project Properties Targets	Misc Engine Version 3.5.21022.8 Schema Version .NET Framework 2.0	
		Schema Version	
	Show Options	dialog before build OK Cancel	

Initially, when you load certain project for the first time (or create a new project), the schema version is set to the same value as Initial Schema Version (see above). When changed, the schema version value is stored in the project configuration file (see Project Configuration section above), and hereafter the project loaded will be edited and built according to the schema version set (regardless of the initial schema version setting).

10.3 Toolset Version Configuration

In MSBuild 3.5, 4.0 and 4.5 projects, it is possible to change the build toolset version. To do so, one ought to select Project element and set the desired toolset version in ToolsetVersion property.

	Properties Elements Raw 3	XML Help
	2 ↓ 2 ↓	
Ξ	General	
	ElementType	Project
	Imported	False
Ξ	Project Specific	
	DefaultTargets	
	InitialTargets	
	ToolsVersion	2.0

10.4 MSBuild 3.5 Editor Support

The following features are available in the editor for MSBuild 3.5 projects only:

- ToolsetVersion attribute is available for root Project element
- ItemDefinitionGroup element can be added under root Project element
- **Remove** attribute is available for Item elements
- ItemGroup/PropertyGroup elements can be added under Target element

• Tasks from Microsoft.Build.Tasks assembly correspond to MSBuild 4.5 version

10.5 MSBuild 4.0 Editor Support

The following features are available in the editor for MSBuild 4.0 projects only:

- **BeforeTargets** and **AfterTargets** attributes are available for Target element.
- Inline tasks can be defined right in the project file under **UsingTask** element. You do not have to create a separate assembly to host the task.
- Tasks from Microsoft.Build.Tasks assembly correspond to MSBuild 4.5 version.

10.6 MSBuild 4.5 Editor Support

The following features are available in the editor for MSBuild 4.5 projects only:

- **VisualStudioVersion** attribute is available for root Project element. By using this attribute you can specify the version of Visual Studio under which this project should be considered to be running.
- **TreatAsLocalProperty** attribute is available for root Project element. By using this attribute you can prevent specific command-line properties from overriding property values that are set in a project or targets file.
- **KeepMetadata** attribute is available for Item element. This attribute specifies the metadata for the source items to add to the target items.
- **RemoveMetadata** attribute is available for Item element. This attribute specifies the metadata for the source items to not transfer to the target items.
- **KeepDuplicates** attribute is available for Item element. This attribute specifies whether an item should be added to the target group if it's an exact duplicate of an existing item.
- The **ContinueOnError** attribute of a Task can contain any of the following values: WarnAndContinue (or true), ErrorAndContinue, and ErrorAndStop (or false).
- Tasks from Microsoft.Build.Tasks assembly correspond to MSBuild 4.5 version.

11. Application Update

You can check for application new releases by choosing Check for Updates menu item from Help menu.



By pressing Install button, application will download new version and start the installer.

Also application can automatically check for updates if *Check for Updates Automatically* options in **Tools** \rightarrow **Options** menu, on **Application Configuration** tab is set to true. You can change the period of updates checks by setting *Period (days)* option.

12. Walkthroughs

11.1 Walkthrough: Creating MSBuild project from scratch

In this walkthrough you will create basic MSBuild project. This project will remove all object (*.obj) and precompiled header (*.pch) files recursively from given folder; if the folder is not passed as a parameter, the folder will default to the project location folder (such script may be useful to remove temporary files from C++ build folders).

Basic knowledge of MSBuild schema is assumed for this walkthrough.

- 1. Create new empty project
 - Click on File->New menu or Standard toolbar New button whereupon new project will be created



2. Add property group

Project Tree New Project

- Select project tree element in project tree by clicking on it
- Right click on project element and choose Add PropertyGroup



- 3. Add property (to specify folder to clean up)
 - Select property group element in project tree view by clicking on it
 - Right click on PropertyGroup element and click on Add Property

Project Tree	
🖃 🛛 😫 Project	
	tvGroup
· · · · · · · · · · · · · · · · · · ·	Add Property
×	Remove PropertyGroup
1	Move Up
1	Move Down

- 4. Assign condition and default value to the property (so default will be supplied if no external value is passed)
 - Select Propertyl element in project tree by clicking on it
 - On Properties tab, click on Check for empty value button on the tab toolbar

Project Tree 🛛 📮 🗙	Properties Elements Raw XM	. Help
🖃 😫 Project 🖃 🐺 PropertyGroup	📜 🛃 🔍 + 🗜 🌆	
🔤 🔷 Property1	🗆 General	
	Condition Check for Empty Value	'\$(Property1)' == "
	ElementType	Property
	Imported	False
	Name	Property1
	Property Specific	
	Value	

- On Properties tab, set Value property to "\$(MSBuildProjectDirectory)"

Properties Elements Ra	w XML Help			
1 2↓ <u>3</u> + <i>f</i> e <i>f</i> a				
General				
Condition	'\$(Property1)' == "			
ElementType	Property			
Imported	False			
Name	Property1			
🗆 Property Specific				
Value	\$(MSBuildProjectDirectory)			

- 5. Add item group
 - Select project element in project tree view by clicking on it
 - Right click on project element and click Add ItemGroup



- 6. Add item (to specify files to be deleted by the project)
 - Select item group element in project tree view by clicking on it
 - Right click on ItemGroup and click Add Item



- Select element item in project tree view by clicking on it
- On Properties tab, set Include property to "\$(Property1)***.obj; \$(Property1)***.pch"

Project Tree	Ψ×	Properties Elements Ra	w XML Help
🖃 😫 Project 🖃 🐉 PropertyGroup		<u>₿</u> 2↓ 3, +	
🔤 🕼 🥥 Property1		🗆 General	
🖃 📲 ItemGroup		Condition	
i 🗈 Item		ElementType	Item
		Imported	False
		Name	Item
		🗆 Item Specific	
		Exclude	
		Include	
		Exclude Include	

Set Include property to "\$(Property1)***.obj; \$(Property1)***.pch; \$(Property1)***.ncb"

Properties Elements Raw XML Help	• ×
2↓ 2↓ -	
🗆 General	
Condition	
ElementType	Item
Imported	False
Name	Item
🖂 Item Specific	
Exclude	
Include	\$(Property1)***.obj; \$(Property1)***.pch;

- 7. Add target
 - Select project element in project tree view by clicking on it
 - Right click on Project element and click Add Target



- 8. Add Delete task to the target
 - Select target element in project tree view by clicking on it
 - Right click on Target element and click Add Task



- Select Delete task in Add Task dialog list
- Click Add button
- Click Close button

Add Task	<u>×</u>
ALLA	Tasks 🔺
	AL
All Tasks	AppDomainIsolatedTaskExtension
	AspNetCompiler
	AssignCulture
	AssignProjectConfiguration
MSBuild	AssignTargetPath
Tasks	CallTarget
	CombinePath
<u> 1</u>	Сору
T	CreateCSharpManifestResourceName
Team Build	Createltem
Tasks	CreateManifestResourceName
	CreateProperty
-42	CreateTemporaryVCProject
Imported	CreateVisualBasicManifestResourceName
Tasks	Csc
	Delete
	Error
L anal Taska	Exec
LUCALLASKS	FindUnderPath
	FormatUrl
<u>∽</u> €	FormatVersion
→ ☆	GenerateApplicationManifest
Custom	GenerateBootstrapper
Tasks	GenerateDeploymentManifest
	GenerateManifestBase
,	
	Add Close

9. Assign Delete task input parameters

- Select task element in project tree view by clicking on it

Project Tree 🛛 📮 🗙		Properties Elements Raw	XML Help
🖃 🐴 Project		≣∳↓ @	
PropertyGroup		General	
🖃 🔐 ItemGroup		Condition	
Item (\$(Property1)*		ContinueOnError	False
🖃 🗠 📥 Target1		ElementType	Task
🗤 🚜 Delete		Imported	False
		Name	Delete
		Input Parameters	
		*Files	
		TreatErrorsAsWarnings	
		Output Parameters	
	Ð	DeletedFiles	

- On Properties tab, set Files property to "@(Item)"

Properties Elements Raw XML Help				
2↓ 2↓ -				
🗆 General				
Condition				
ContinueOnError	False			
ElementType	Task			
Imported	False			
Name	Delete			
🖂 Input Parameters				
*Files	@(Item)			
TreatErrorsAsWarnings				
🖂 Output Parameters				
DeletedFiles				
DeletedFiles				

- 10. Assign Delete task output parameters
 - Select task element in project tree view by clicking on it
 - Right click on Delete task element and click DeletedFiles from Add Output submenu



- Select task output parameter element in project tree view by clicking on it
- On Properties tab, set PropertyName property to "deleted"



- 11. Add Message task (to print the deleted files list)
 - Right click on Target element from project tree and choose Add Task
 - Select Message task in Add Task dialog tasks list
 - Click Add button
 - Click *Close* button

Project Tree 🛛 📮 🗙	Properties Elements	Raw XML Help			
Project	2↓ ☆ -				
Property1	🖂 General				
	Condition				
Item (\$(Property1)***.ob)	ContinueOnError	False			
🖃 🗠 🛔 Target1	ElementType	Task			
	Imported	False			
+ Output (DeletedFiles)	Name	Message			
	🖂 Input Parameters				
	Importance	normal			
	Text				

- 12. Assign Message task input parameters
 - Select task element in project tree view by clicking on it
 - On Properties tab, set Text property to "\$(deleted)"

Properties Elements Raw XML H	elp
2↓ 2↓ -	
∃ General	
Condition	
ContinueOnError	False
ElementType	Task
Imported	False
Name	Message
∃ Input Parameters	
Importance	normal
Text	\$(deleted)

13. Set project default target

- Select project element in project tree view by clicking on it
- On Properties tab, set *DefaultTargets* property to "Target1"

Project Tree 🛛 📮 🗙	Properties Elements Rav	XML Help
E 😫 Project	2↓ 2↓ -	
🎰 🌒 Property1	🖂 General	
🚊 🖳 🚰 ItemGroup	ElementType	Project
Item (\$(Property1)***.ob)	Imported	False
🖻 🗠 📩 Target1	🗆 Project Specific	
	DefaultTargets	Target1
	InitialTargets	
······ 🛶 Message	System Properties	
	MSBuildProjectDirectory	C:\Documen
		i

14. Save project file

- Click *File->Save* menu or Standard toolbar *Save Project* button



Upon completing the walkthrough, the following MSBuild project was created:

```
<Project xmlns="http://schemas.microsoft.com/developer/msbuild/2003"

DefaultTargets="Target1">

<PropertyGroup>

<PropertyI Condition="'$(Property1)' == ''">$(MSBuildProjectDirectory)</Property1>

</PropertyGroup>

<ItemGroup>

<Item Include="$(Property1)\**\*.obj;$(Property1)\**\*.pch;" />

</ItemGroup>

<Target Name="Target1">

<Delete Files="@(Item)">

<Output TaskParameter="DeletedFiles" PropertyName="deleted"/>

</Delete>

<Message Text="$(deleted)" />

</Project>
```

11.2 Walkthrough: Building and using build logs

In this walkthrough, you will run build on existing build project and review logs of the build process; the project created in "Creating project from scratch" walkthrough will be used. To review the project build logic and logs it is advisable also to create a temporary folder containing files with object (*.obj) or precompiled header (*.pch) extensions.

Basic knowledge of MSBuild schema and build logic is assumed for this walkthrough.

- 1. Load saved project
 - Click File->Open menu or Standard toolbar Open Project button and select the existing project

File	Edit V	'iew	Build	Tools H
	New	Ctrl	+N	
2	Open	Ctrl-	+0	
	Close			
	Save	Ctrl	+S	
	Save As			
	Reload			
	Recent Pro	ojects	+	
	E×it			
				_



_

- 2. Specify external parameters for the build
 - Click Build->Build Options menu or Build toolbar Build Options button



- Switch to Project properties tab by selecting Project properties shortcut on the left pane of Build Options dialog
- Add new record to table: Name = "Property1", Value = "path" (where path is set to the path containing obj and pch files)

Bu	uild Options			2
		<u>*</u> ×		
	Engine	Name	V	Value
	Fropentes	Property1		some/other/path
	1			
	Enviroment Properties			
	Project Properties			
	∰ Targets			

3. Build project

- Click Build->Build Project menu or Build toolbar Build Project button





- 4. Review logs (errors, detailed, console)
 - Build Console log
 - Open log window (View->Log window menu or View toolbar Show Log window button)



	2				
Show Log Window Elements					
₿ 2↓	127				

- Switch to Build Console log by choosing corresponding item from log window combobox

Log Window			
Show Log:	Edit Project Log	-	🛒 🕕
N N	Edit Project Log Build Console Log Build Errors/Warnings Build Detailed Log		

- Log contains all build information (similar to MSBuild engine command-line console log)

Log Window		
Show Log:	Build Console Log	- 🛒 🕕
Build succes OWarning OError(s) Time Elapse	eded. g(s) d 00:00:00.07	

- Build Errors/Warning log
 - Switch to Build Errors/Warnings by choosing corresponding item from log window combobox. If project build succeeded this log will be empty

Log Window							
Show Log:	Show Log: Build Errors/Warnings 🔹 式 🕕						
Event Type		Sender	File	Line	Column	Message	

- If there were errors/warning during build log contains information about them (filename, line, column and message if provided by MSBuild engine), and helps to fix the problem using provided information

Log Window							
Show Log:	Show Log: Build Errors/Warnings 🔽 🛒 🕕						
Event Type	Sender	File	Line	Column	Message		
🚹 Warning	Warning	g C:\Documents	and 0	0	The project file		
🔇 Error	Сору	C:\Documents	: and 0	0	Unable to copy		
🔇 Error	Сору	C:\Documents	: and 0	0	Unable to copy		
🔇 Error	Сору	C:\Documents	and 0	0	Unable to copy		
🔇 Error	Сору	C:\Documents	: and 0	0	Unable to copy		

- Build Detailed Log
 - Switch to Build Detailed log by choosing corresponding item from log window combobox
 - Log contains elements build order information and status of each element
 - By choosing value from Status combobox located on the window toolbar, the list can be filtered by specific status

Status:	Failed 🔻
Time	All Succeed Failed Skipped Targets Skipped Tasks

- Double-clicking on selected element will select the element in the project tree view. Note: First occurrence of element with same name and type in project tree will be selected

Project Tree # × Project PropertyGroup Property1 ItemGroup ItemGroup Item(\$(Property1))****.ot Item (\$(Property1))****.ot Itemeter Item (\$(Output (DeletedFiles)) Itessage	Properties Elements Project Specific DefaultTargets InitialTargets System Properties MSRuidDrojectDirectory	Raw XML Help Project False Target1
Log Window		
Show Log: Build Detailed Log 🔹 🛒 🕕) Status: All	•
Element	Time File	D
🖃 🚑 Project 🛛 🥑	00:00:00.264 1.proj	
🔳 🛓 Target1 🛛 🖉	00:00:00.140 1.proj	

11.3 Walkthrough: Debugging and using Globals/Autos windows

In this walkthrough, you will debug an existing Visual Studio C# project, set breakpoints for particular Tasks and monitor Property values using Globals/Autos windows.

Basic knowledge of MSBuild schema and build logic is assumed for this walkthrough.

1. Open valid C# project

- Click File->Open menu or Standard toolbar Open Project button and select C# project file to open

F	ile	Edit	View	Build	Tools He
	1	New	Ctr	l+N	
	2	Open	Ctr	l+0	
		Close			
L	1	Save	Ctr	·l+s	
		Save As.			
		Reload			
		Recent P	rojects	•	
		Exit			

- 2. Open Debug, Globals, Autos windows
 - Click View -> Debug Window menu
 - Click View -> Globals Window menu
 - Click View -> Autos Window menu

Viev	/ Build	Tools	Help	
.	Project Tr	ree		
	Log Wind	DW		
9	Debug Wi	indow		
<u></u>	Globals W	'indow		
	Autos Wir	ndow		

- 3. Navigate to Debug window and toggle breakpoint on GetFrameworkPaths and CoreCompile Targets.
 - Right mouse click on Target element and click on Toggle Breakpoint menu item



4. Start debugging

- Click *Build->Debug* menu or Build toolbar *Debug* button

Buil	d <u>T</u> ools <u>H</u> elp	
	Debug	F5
≡⊳	<u>B</u> uild Project	Ctrl+F5
Ē	Stop Building Project	Shift+F5
• 1	Debug Step	F11
E 🎲	Build Options	



- 5. Perform three debug steps until PrepareForBuild Target becomes highlighted
 - Click Build->Debug Step menu or Build toolbar Debug Step button

Bui	ld	<u>T</u> ools	<u>H</u> elp		
	C	ebug)		F5	
	E	uild Proje	ct	Ctrl+F5	
	ŝ	itop Buildi	ng Project	Shift+F5	
9 3	C	ebug Ste	P	F11	
	Е	uild <u>O</u> ptic	ns		
		9 <u> </u>			
		Deb	ug Step (F1	1)	

- 6. View the Targets/Tasks build order on Debug diagram.
 - Note that Targets/Tasks highlighted with green were executed in the build; Targets/Tasks highlighted with yellow were skipped

Elements Raw XML Help Debug Log	• ×
🔍 🔍 🚂 🎒 🔩	
CheckForInvalidConfigurationAndPlatform	
BuildOnlySettings	
● *GetFrameworkPaths → C CetFrameworkPath → C CetFrameworkSdkPath	
PrepareForBuild > 4% FindAppConfigFile > 4% MakeDir	
	•

- 7. View .NET Framework paths detected by GetFrameworkPath task on Globals window
 - Select Items to be displayed in drop-down on Globals window



 Items that are updated on GetFrameworkPaths Target execution are highlighted with red on Globals window

Globals	•	×
Show: Items - Filter:		
Name	Value	
$_CombinedTargetFrameworkDirectoriesItem$	C:\WINDOWS\Microsoft.NET\Framework\v3.0	
_CombinedTargetFrameworkDirectoriesItem	C:\WINDOWS\Microsoft.NET\Framework\v3.5	
_CombinedTargetFrameworkDirectoriesItem	C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727	
_TargetFramework20DirectoryItem	C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727	
_TargetFramework30DirectoryItem	C:\WINDOWS\Microsoft.NET\Framework\v3.0	
_TargetFramework35DirectoryItem	C:\WINDOWS\Microsoft.NET\Framework\v3.5	
_TargetFrameworkSDKDirectoryItem	C:\Program Files\Microsoft SDKs\Windows\v6.0A\	
_ApplicationManifestFinal	bin\Debug\ConsoleApplication5.exe.manifest	
_DebugSymbolsIntermediatePath	obj\Debug\ConsoleApplication5.pdb	
		- 1000000

8. Resume project execution until Csc Task is reached

- Click Build->Debug menu or Build toolbar Debug button

Build	d <u>T</u> ools <u>H</u> elp	
	Debug	F5
≡⊳	<u>B</u> uild Project	Ctrl+F5
F 🔲	Stop Building Projec	t Shift+F5
9	Debug Step	F11
	Build Options	

=▶ 🕨 📲 💷 🎲 Help Debug (F5) og

- CoreCompile Target will become selected on Debug window



Click Build->Debug Step menu or Build toolbar Debug Step button
 Click Build->Debug Step menu or Build toolbar Debug Step button



- 9. View options and files to be passed to C# compiler on Autos window
 - Navigate to Autos window to view Properties and their values passed to Csc Task

Autos	•	×
Show: Properties	- Filter:	
Name	Value	
CscToolExe		
CscToolPath		
DebugSymbols	true	
DebugType	full	
DefineConstants	DEBUG;TRACE	
DelaySign		
ErrorReport	prompt	
FileAlignment	512	
GenerateFullPaths		
KeyContainerName		•

- To view Items passed to Csc task, select Items in drop-down on Autos window



Autos	• X	
Show: Items	▼ Filter:	
Name	Value	
Compile	Properties\AssemblyInfo.cs	
Compile	Program.cs	
IntermediateAssembly	obj\Debug\ConsoleApplication5.exe	
ReferencePath	C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\System.dll	
ReferencePath	C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\System.Xml.dll	
ReferencePath	C:\Program Files\Reference Assemblies\Microsoft\Framework\v3.5\System.Xml.Linq.dll	
ReferencePath	C:\Program Files\Reference Assemblies\Microsoft\Framework\v3.5\System.Core.dll	
ReferencePath	C:\Program Files\Reference Assemblies\Microsoft\Framework\v3.5\System.Data.DataSetExtensions.dll	
ReferencePath	C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\System.Data.dll	

11.4 Walkthrough: Building Visual Studio C# project different configurations

In this walkthrough, you will build an existing Visual Studio C# project, change build configuration from default (Debug) to Release and review logs of the build process.

Basic knowledge of MSBuild schema and build logic is assumed for this walkthrough.

- 2. Open valid C# project
 - Click File->Open menu or Standard toolbar Open Project button and select valid C# project to open





3. Build project

Click Build->Build Project menu or Build toolbar Build Project button





- 4. Review logs (error, detailed, console)
 - Please see Building and using build logs walkthrough for details on reviewing logs
- 5. Change configuration from Debug to Release for the build
 Click *Build->Build Options* menu or Build toolbar *Build Options* button



I 🕮 😭	
Prope Build Options ents Raw XML	_

 Switch to Project properties page by selecting Project properties shortcut on the left pane of Build Options dialog Add new entry to table: Name = "Configuration", Value = "Release" (the new entry sets property Configuration value to Release for the build)

Bu	Build Options					
	Engine Properties Enviroment Properties	°⊒ 🗙 Name	Value			
		Configuration				
	Project Properties					
	💒 Targets					
	Show Option	s dialog before build		Cancel		

- 6. Build Release configuration
 - Execute step 2 again. Reviewing the logs will show that build was performed for Release configuration