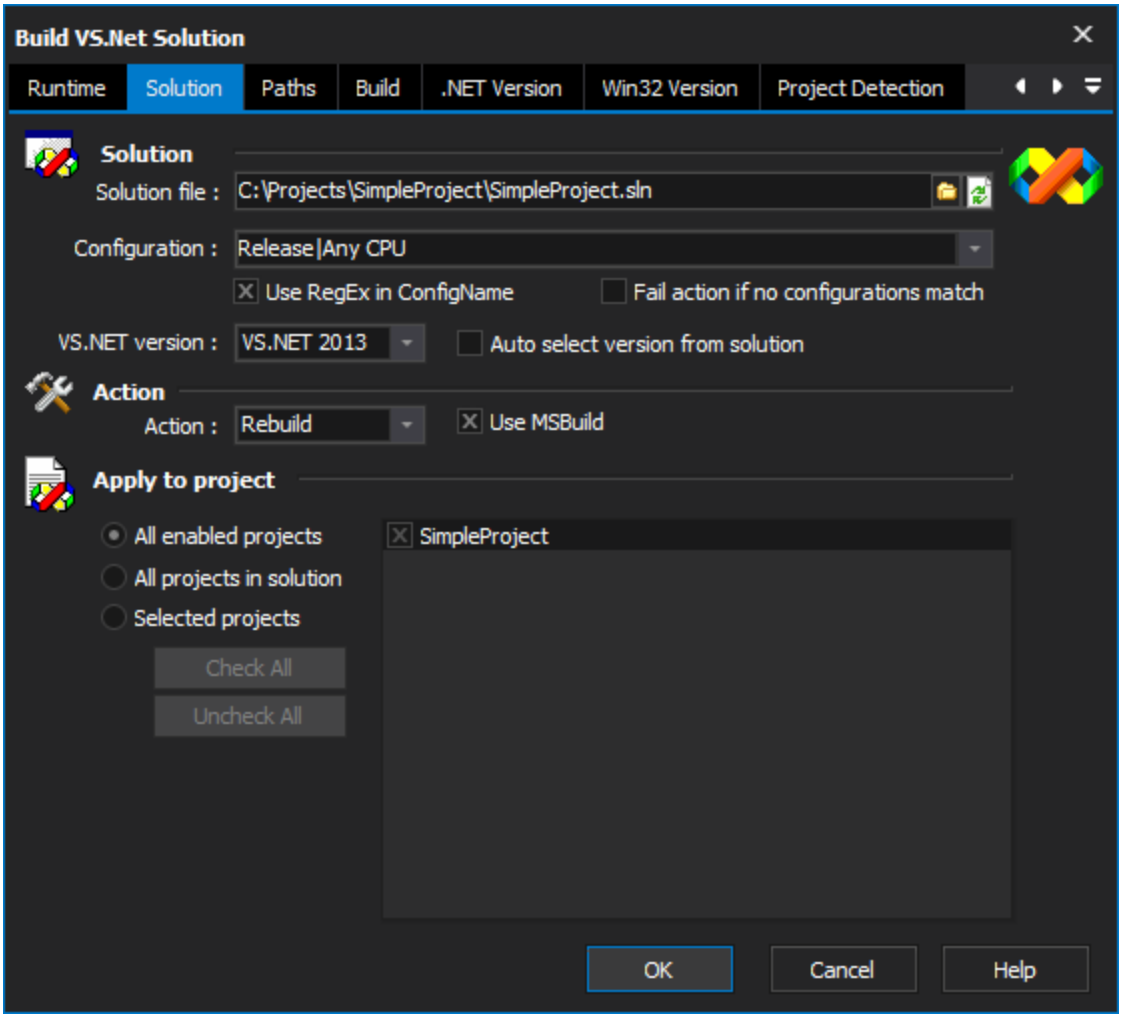
Visual Studio .NET Action

This action provides the ability to build Visual Studio .NET solutions using FinalBuilder. You can choose to build, rebuild or clean the whole solution or selected projects in the solution. You can also choose which Solution Configuration to use.



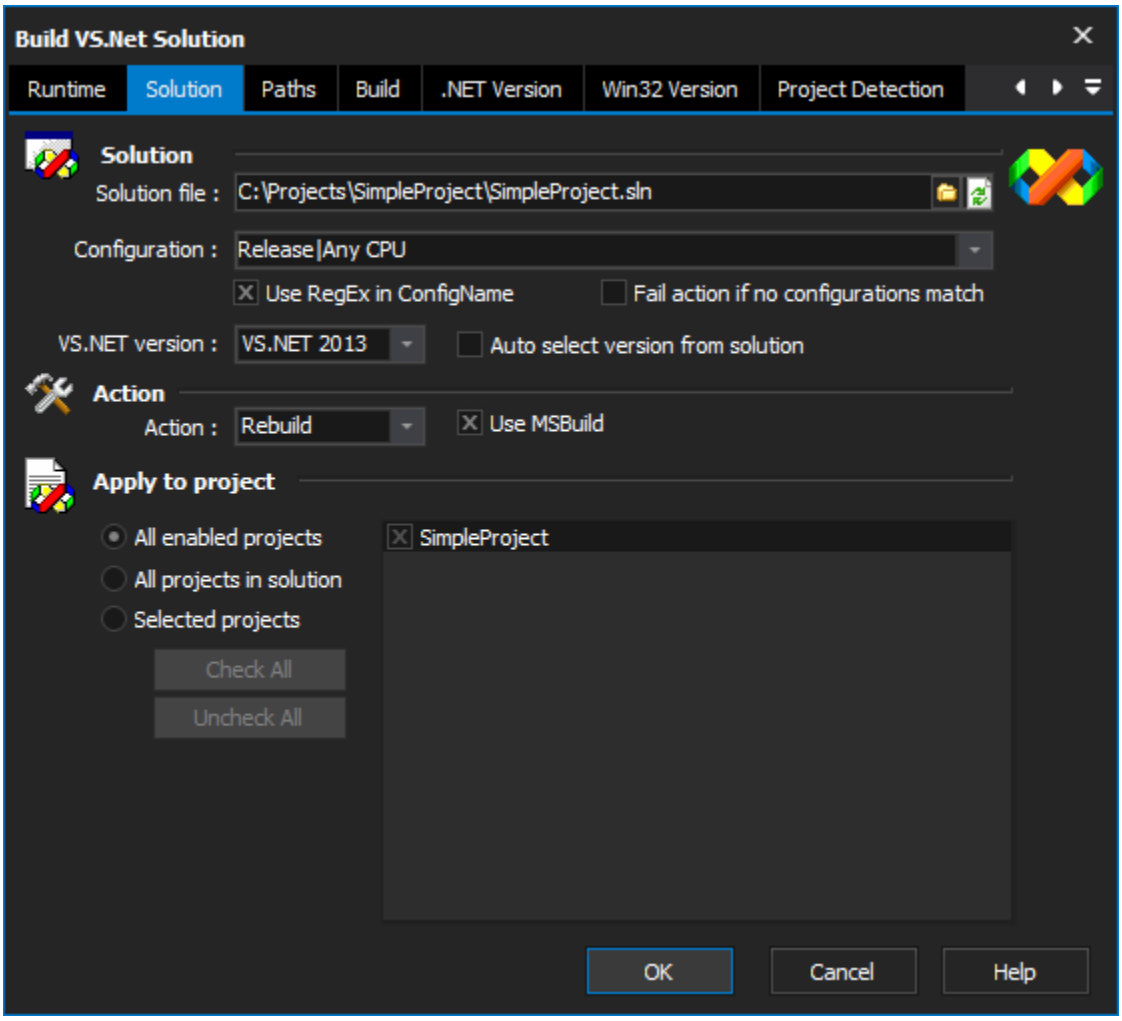
On this page:

Solution

Solution File

The path to the solution file to build. If the solution file is found at design-time, the configuration and project lists will be automatically populated.

Click the refresh button (as shown) to reload the solution and refresh the configuration, version and project names.



Configuration

Choose the solution configuration to build. Available configurations are automatically loaded from the solution file and shown in the dropdown list.

If the solution file cannot be found, the dropdown shows the default Visual Studio.NET configurations.

Use RegEx in config name

Allows the specification of a regex statement for the configuration. The configuration will be interpreted by splitting the text into two parts. The parts are split by the pipe symbol.

For example a configuration value of of "Release|x64" will split into two regular expressions of "Release" and "x64". The first part is tested again the configurations found in the selected projects, while the second part is tested against the platform. When a target is found that has a match for the configuration and platform regex it will be added to this list of targets to build for the project.

Examples:

Configuration Text	Configuration Regex	Platform Text	Example Targets Matched
Release|x86	Release	x86	Release - x86 Release And Deploy - x86
Release*|*	Release*	*	Release - x86 Release - x64 ReleaseAndDeploy - x64 ReleaseToTesting - x64

Deloy|	*Deploy	x*	ReleaseAndDeploy - x64 ReleaseAndDeploy - x86 Deploy - x64 Deploy - x86

Fail action if no configurations match

Select this option if you wish to have the action fail when no configurations match the supplied configuration value.

VS.NET Version

Specify the version of Visual Studio to build the solution. A solution file can only be built by the version of Visual Studio that created it.

The correct version is automatically chosen in the solution file is loaded at design-time.

Auto select version from solution

Select this option if the version of the compiler should be taken from the supplied solution file.

Action

Action

Choose the action (Build, Rebuild, Clean or Deploy) to perform. These are all the equivalent of taking the same action in the Visual Studio IDE.

Use MSBuild

Solution files created by Visual Studio 2005 and newer can be built by MSBuild. Check the "Use MSBuild" checkbox to build with MSBuild instead of Visual Studio. If you enable this option, you do not need Visual Studio installed on the build machine. Choosing this option also enables a lot of the custom properties of the action, and allows you to set Unloaded Projects (see below for details.)

Apply To Project

You can choose to build all projects in a solution, or only build certain projects.

All Enabled Projects

Selecting this option will build the chosen configuration, as defined in the Configuration Manager in Visual Studio. Only projects with the "Build" flag set in Configuration Manager will be built.

All Projects in Solution

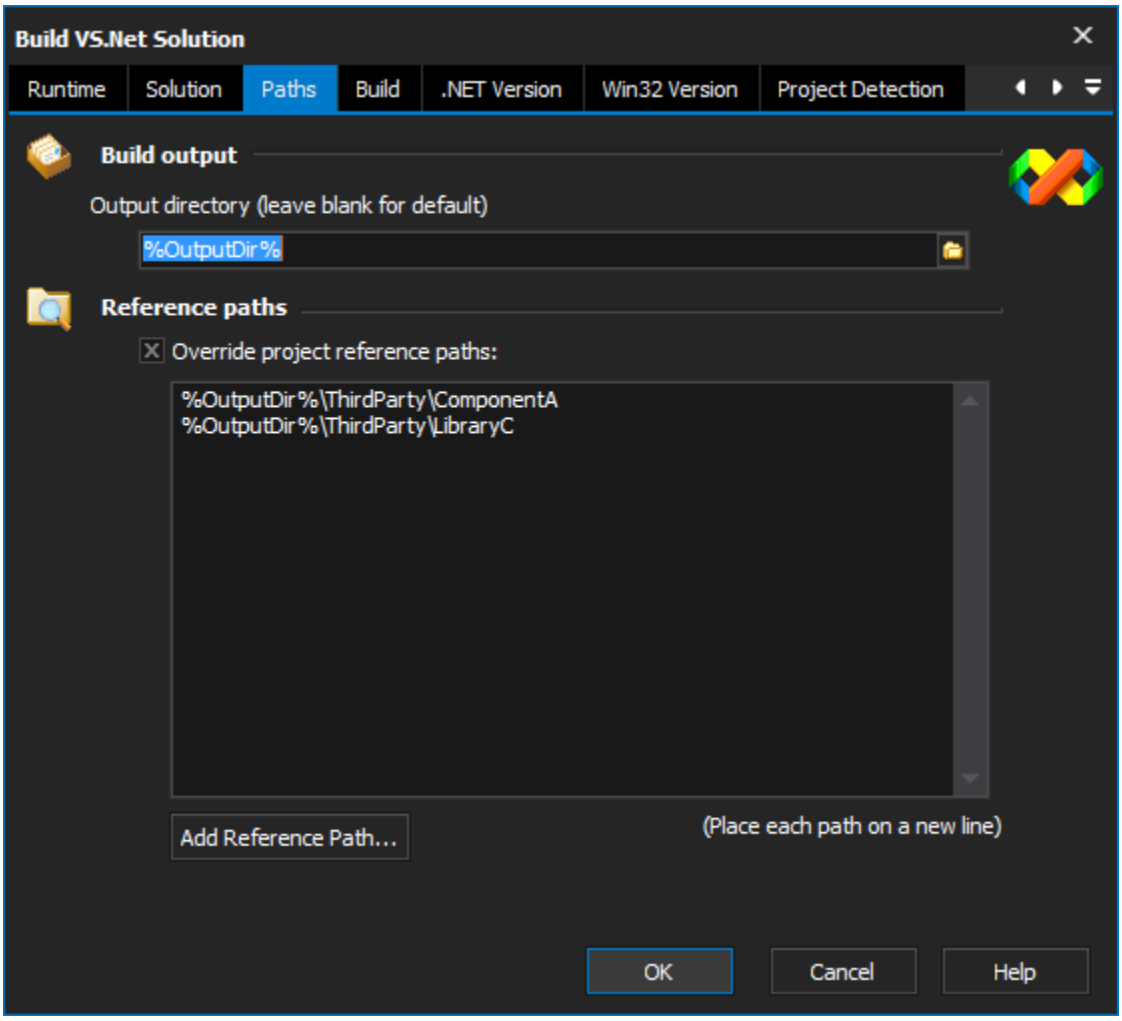
Selecting this option means that every single project in the solution will be built, regardless of whether or not it is set to "Build" as part of the selected configuration.

Selected Projects

Selecting this option allows you to choose which projects to build, using the listbox shown to the right of the option. Only checked projects will be built.

See Unloaded Projects (see below for details.)

Paths



(The options on this page are only available when the "Use MSBuild" option is enabled.)

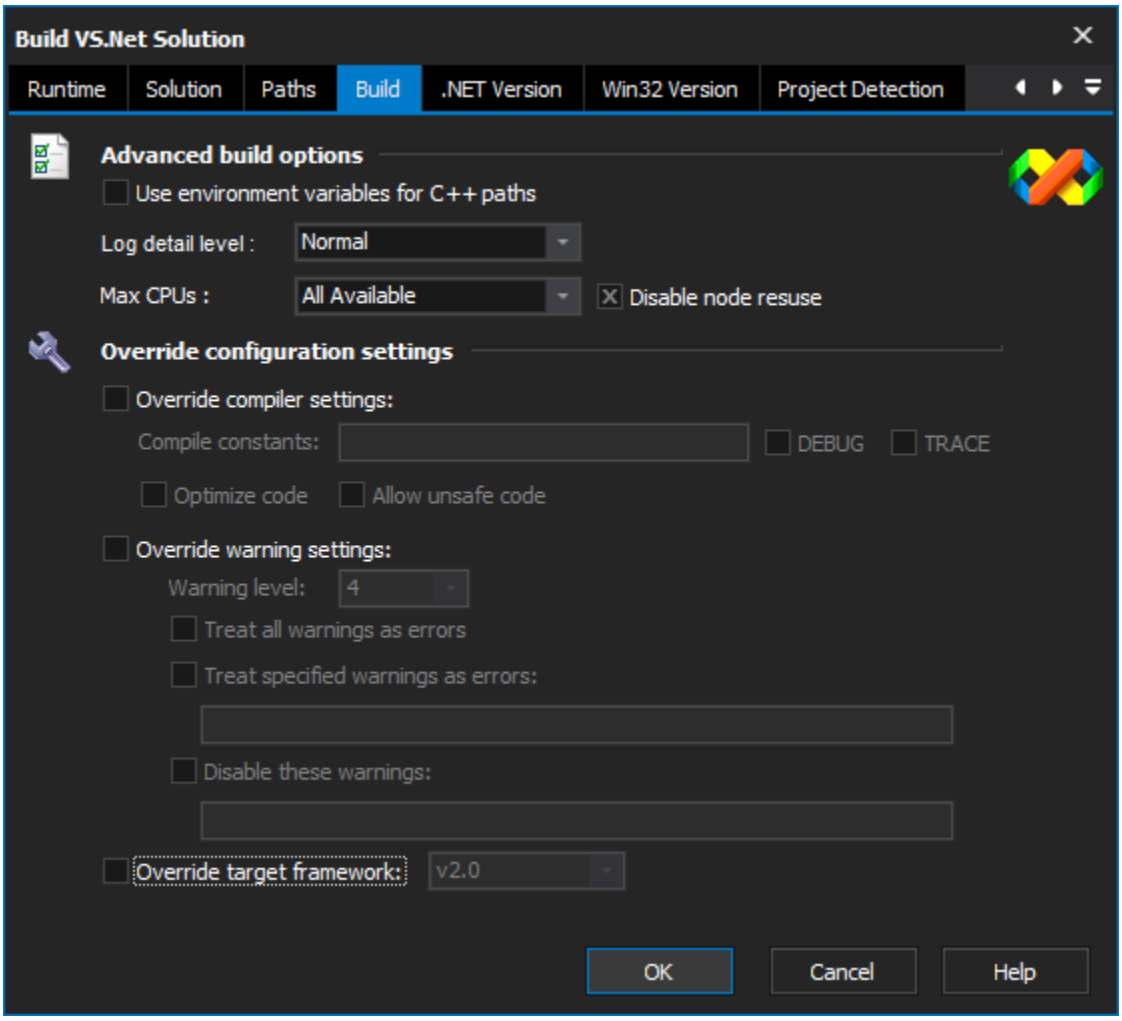
Build Output

Output Directory

Enter an output directory here to override the default output directory for each project in the solution. If this field is left blank, the default output directory will be used.

Reference Paths

If the "Override project Reference Paths" option is enabled, this list of paths will be used to resolve search for assembly references. This list overrides the list of paths set on the "Reference Paths" tab for a Visual Studio project (this list is normally stored in each project's .user file.)



(Apart from "Use Environment Variables...", all of the options on this page are only available when "Use MSBuild" is enabled.)

Build

Advanced Build Options

Use Environment Variables for C++ Paths

When building Visual C++ projects, enabling this option will use the environment variable values PATH, INCLUDE, LIBS, and LIBPATH when starting Visual Studio. This is equivalent to passing the /useenv option to Visual Studio, or the /p:useenv=1 argument to MSBuild.

Log Detail Level

Specify the level of output detail you would like to see in the log. Minimal will only show basic compilation steps. Normal shows all major targets. Detailed and Diagnostic can be used to analyse complex build scenarios.

Max CPUs

This option is only enabled when Visual Studio 2008 is being used for the build. You can limit the maximum number of CPUs to use for building, or set it to "All Available" to use all available CPUs.

Override configuration settings

Override compiler settings

If this checkbox is selected, then all of the following configuration properties will be overriden:

Compile Constants

Specify conditional constants to define. Separate multiple constants with semicolons.

DEBUG & TRACE

Check these boxes to specifically enable these two constants.

Optimize Code

Check this box to enable code optimization.

Allow unsafe code

Check this box to allow unsafe code.

Override Warning settings

If this checkbox is selected, then all of the following Warning-related properties will be overriden:

Warning Level

Specify the level of warning output to display.

Treat All Warnings as Errors

If this option is enabled, any compiler warning will cause the build to fail.

Treat Specified Warnings as Errors

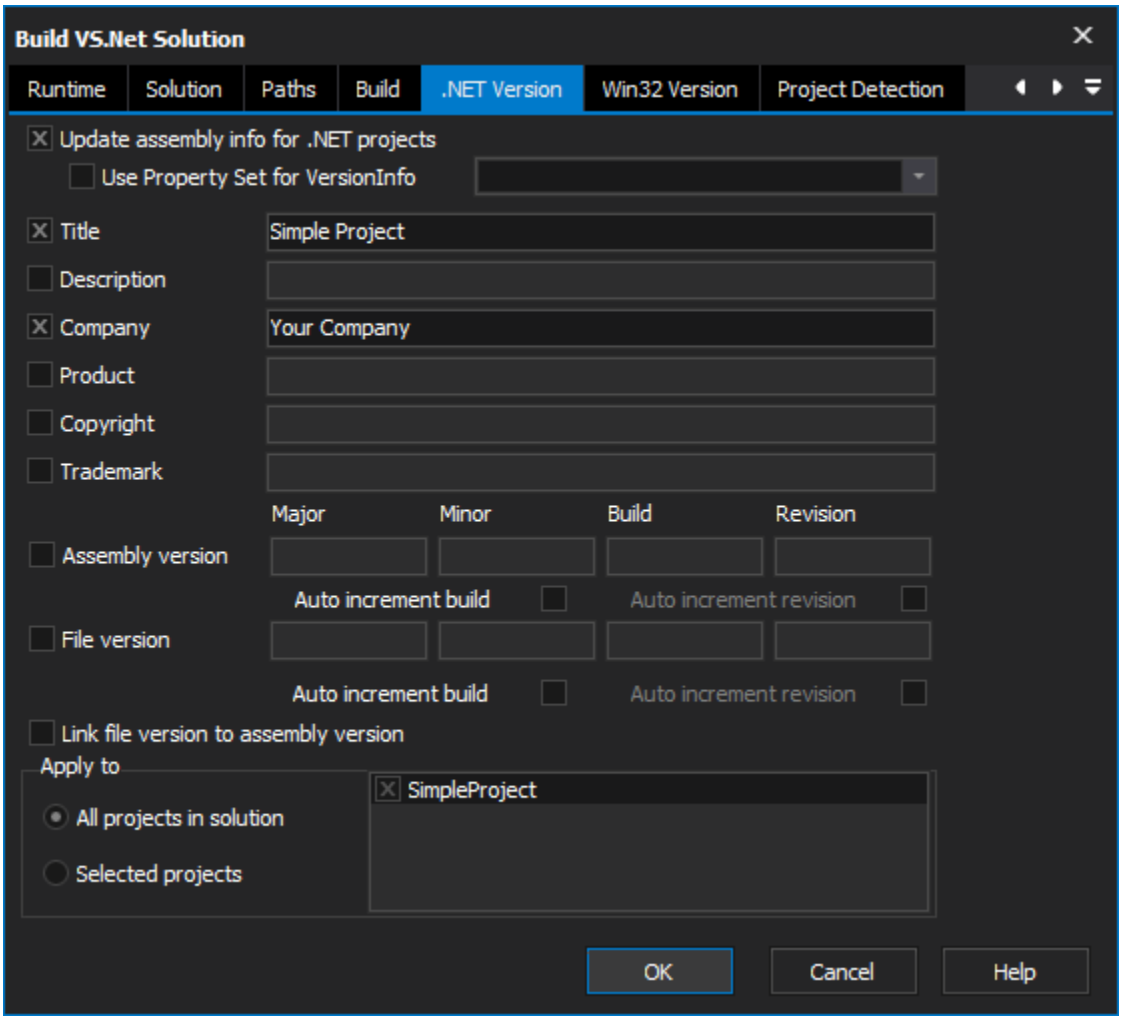
Specify one or more warning numbers to treat as errors. Separate multiple warning numbers with semicolons.

Disable These Warnings

Specify one or more warning numbers to completely ignore. Separate multiple warning numbers with semicolons.

Override Target Framework

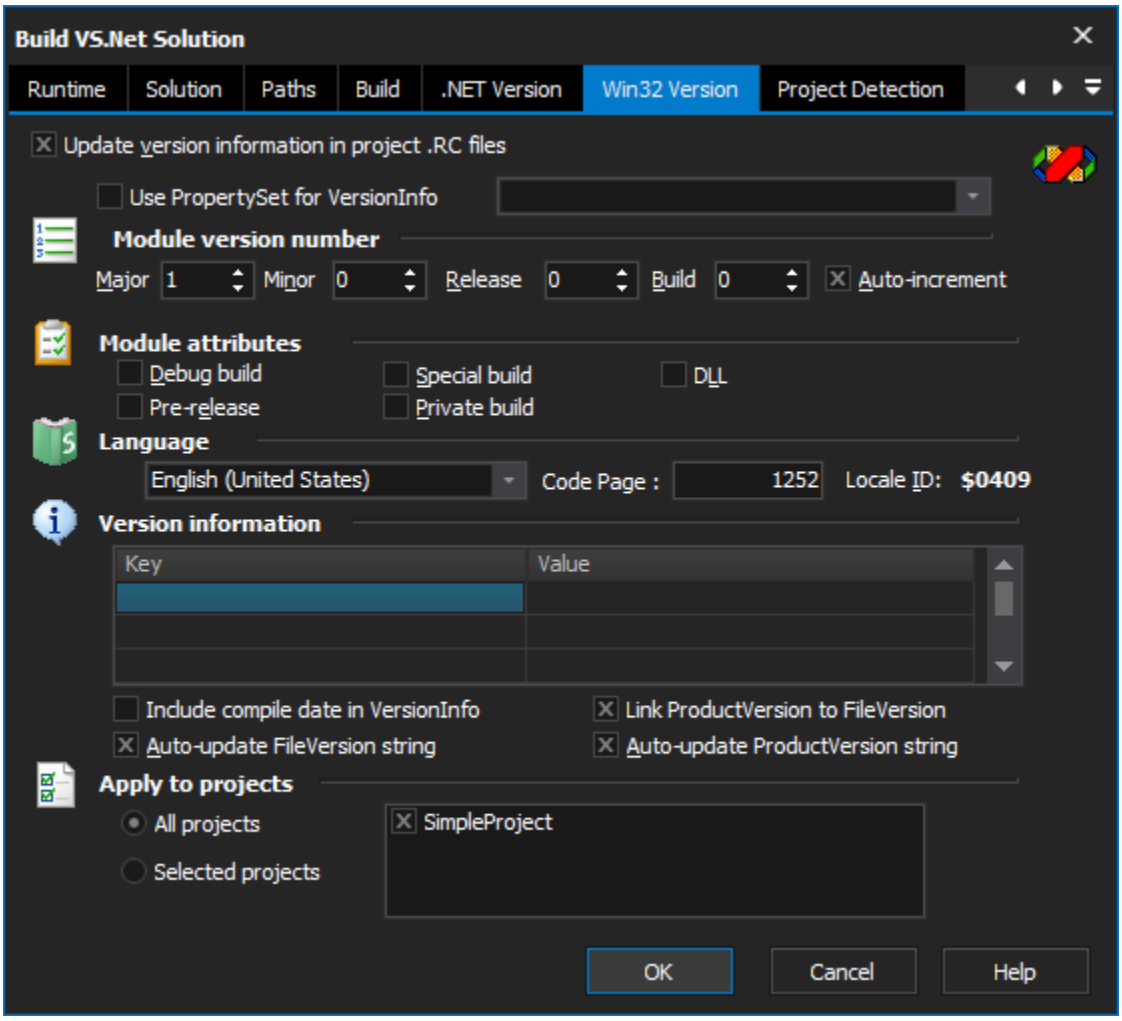
This option is only enabled when using Visual Studio 2008. It allows you to build projects against an earlier version of the .NET Framework.



.NET Version

The .NET Version page allows the action to update AssemblyInfo files in all or selected .NET projects in the solution. This will not work for any ASP.NET Web Sites in the solution.

Specify the details that you would like to set. The "Use PropertySet for Version Info" dropdown allows you to use a dedicated topic.)



The Win32 Version page allows the action to update .RC files in all or selected C++ projects in the solution.

Specify the details that you would like to set. The "Use PropertySet for Version Info" dropdown allows you to use a dedicated topic.)

Update assembly info for .NET projects

Select this option if the assembly information for the solution and its projects should be updated to those supplied on the page.

Use property set for version info

Select this option if a property set should be used to retrieve the relevant version information for the assembly. The property set from which to gather the version information much be supplied before the action is validated.

Title

Includes the supplied title into the assemblies version information.

Description

Includes the supplied description into the assemblies version information.

Company

Includes the supplied company name into the assemblies version information.

Product

Includes the supplied product name into the assemblies version information.

Copyright

Includes the supplied copyright text into the assemblies version information.

Trademark

Includes the supplied trademark text into the assemblies version information.

Link file version to assembly version

Select this option if the file version should be linked to the assembly version in the assembly information page.

Assembly version & File version

Major

The major number for the version information.

Minor

The minor number for the version information.

Build

The build number for the version information.

Release

The release number for the version information.

Auto increment build

Whether to automatically increase the build number before the version information is applied to the assembly.

Auto increment revision

Whether to automatically increase the revision number before the version information is applied to the assembly.

Apply to

All projects in solution

Whether the assembly information is applied to all projects in the solution.

Selected projects

Select to apply the assembly information to only the projects selected.

Version Info

Update version information in project

Whether the linker should include version information into the generated binary or not.

Use Property Set for Version Info

Whether to use a Property Set for the version information of the generated binary or not.

Module version number

Major version

The major version number of the generated binary. Can be set directly, or taken from a supplied Property Set.

Minor version

The minor version number of the generated binary. Can be set directly, or taken from a supplied Property Set.

Release

The release version number of the generated binary. Can be set directly, or taken from a supplied Property Set.

Build

The build version number of the generated binary. Can be set directly, or taken from a supplied Property Set.

Auto-increment build number

Whether to automatically increase the build number portion of the Module Version Number or not. If the number is increased the value is updated into the action. The project will need to be saved so that this persists for the next execution.

Module Attributes

A binary can be marked with a number of different attributes. These are namely

Debug build

Indicates that the project was compiled in debug mode.

Special build

Indicates that the version is a variation of the standard release.

DLL

Indicates that the project includes a dynamic-link library.

Pre-release

Indicates the version is not the commercially released product.

Private build

Indicates that the version was not built using standard release procedures

Language

The primary language of the binary.

Code Page

Determines the default text character set that is used in ANSI source code. This is used by the compiler in working out how to parse multibyte character strings. This typically includes;

- String constants
- Comments
- #error directive
- #define directive

Locale ID

The locale ID which relates to the code page selected.

Version Information

All the version information which will be attached to the binary when its generated. These can either be sourced from the Project Settings, Property Set, or set directly in the dialog.

Include Compile Date in Version Info

Where to update the version information to have the date of the action run included into the version information as the compile date.

Auto-Update FileVersion String

Whether to automatically update the FileVersion string based on the FileVersion major, minor, release, and build numbers.

Link ProductVersion to FileVersion

Whether to link the ProductVersion file in the version information to the FileVersion field. Typically used if the product has only one version number across all parts of the build.

Auto-Update ProductVersion String

Whether to automatically update the ProductVersion string based on the ProductVersion major, minor, release, and build numbers.

Apply to Project

All Projects

Whether to apply Win32 version information to all projects.

Selected Projects

Select to apply Win32 version information to only the selected projects.

Project Detection

Use WMI - specify website number

The website number to use for updating the assembly information.

Use network share - specify share name

The website share for updating the assembly information.

Use WebDAV

The WebDAV username and password for updating the assembly information.

Scripting

In addition to the common scripting properties, this action supports 3 additional script functions

Action.GetProjectID

This function takes the path to a project file, or the name of a project, and returns the internal ID (GUID) for that project. If the project is not found in the current solution, the function returns an empty string. Paths can be absolute, or relative to the solution file.

Action.GetProjectName

This function takes the ID of a project inside the solution, and returns the name (as shown in the Visual Studio IDE.) If the project ID is not found in the current solution, the function returns an empty string.

Action.GetProjectPath

This function takes the ID of a project inside the solution, and returns the project file path (relative to the solution directory.) If the project ID is not found in the current solution, the function returns an empty string.

Dynamically Choosing Which Projects to Build at Runtime

It is possible to dynamically configure the list of projects to build at runtime, using the Action.SelectedProjects property. This property is a variable "MyProjects", which contains a list of project names or paths, it is possible to use script like this in the BeforeAction event of the action:

Action.SelectedProjects = FBVariables.MyProjects;

Names are automatically converted to project IDs when the value is set. If the name does not represent a valid project, the entry is not added

Solutions Which Contain Unloaded Projects

It is possible to unload a project in Visual Studio by right-clicking in Solution Explorer and selecting "Unload Project". Unloaded projects build differently, depending on if you have chosen "Use MSBuild" or not.

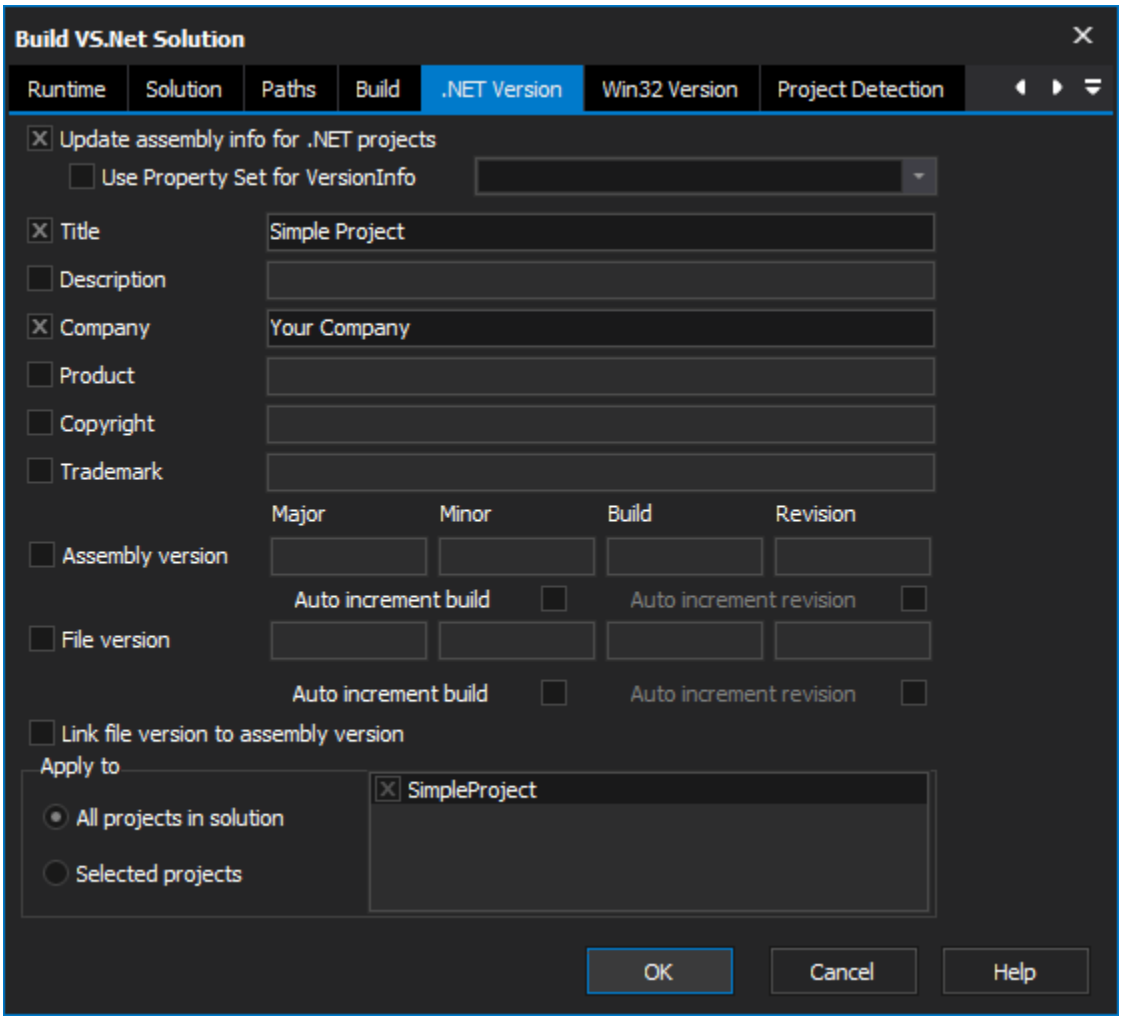
With "Use MSBuild" enabled, unloaded projects behave exactly the same as loaded projects. If you choose to build "All Projects", then all projects (including unloaded projects) will build. If you choose to build "Selected Projects", then all the selected projects will build.

Without "Use MSBuild" enabled, unloaded projects will never be built, even if they are explicitly selected.

Extra MSBuild Properties

It is possible to pass custom MSBuild Properties to the build process when "Use MSBuild" is selected. This allows you to set build properties which are not supported natively by the action.

To set these properties, select the action and click on the Properties tab. Under the MSBuild section is a property called "Extra MSBuild Properties":



Click the ellipsis button to edit the list of extra properties. Place each property on a new line. Properties take the form <Name>=<Value>.

.NET Core Project Assembly Info

.NET Core projects by default do not include an AssemblyInfo.cs/vb file, instead they use the package info inside the newer project format. This can however be changed to allow FinalBuilder to update the assemblyinfo just like it did before .net core

1) Add a properties folder to the project (VS2017 will automatically change the folder icon), and then add an AssemblyInfo.cs/vb file to that folder. This file then becomes the source of assembly info for the project, and FinalBuilder is able to find and update the file

2)Edit the .csproj /.vbproj file and add a GenerateAssemblyInfo element with a value of false :

```
<PropertyGroup>
<OutputType>Exe</OutputType>
<TargetFramework>netcoreapp2.0</TargetFramework>
<AssemblyVersion>1.2.3.4</AssemblyVersion>
<FileVersion>1.2.3.4</FileVersion>
...
<GenerateAssemblyInfo>false</GenerateAssemblyInfo>
</PropertyGroup>
```

Also, either add a new Item Group

```
<ItemGroup>
  <None Include="Properties\AssemblyInfo.cs" />
</ItemGroup>
```

or add the None entry to an existing ItemGroup.