This documentation is archived and is not being maintained. Recommended Version

Macros for Build Commands and Properties

Visual Studio 2012

You can use these macros anywhere in a project's **Property Pages** dialog box where strings are accepted. These macros are not case sensitive.

To display the currently available macros, in the column to the right of a property name, click the drop-down arrow. If **Edit** is available, click it and then in the edit dialog box, click **Macros**. For more information, see the **Specifying User-Defined Values** section of Property Pages (C++).

Macros that are marked "Deprecated" are no longer used or have been replaced by an equivalent item metadata macro (**%(***name***)**). Macros that are marked "Deprecated; migrated" are also deprecated. And in addition, if the project that contains the macro is migrated from Visual Studio 2008, Visual Studio converts the macro to the equivalent Visual Studio 2010 macro.

Macro	Description
$(RemoteMachine)	Set to the value of the **Remote Machine** property on the Debug property page. See Changing Project Settings for a C/C++ Debug Configuration for more information.
$(Configuration)	The name of the current project configuration (for example, "Debug").
$(Platform)	The name of current project platform (for example, "Win32").
$(ParentName)	(Deprecated.) Name of the item containing this project item. This will be the parent folder name, or project name.
$(RootNameSpace)	The namespace, if any, containing the application.
$(IntDir)	Path to the directory specified for intermediate files relative to the project directory. This path should have a trailing slash. This resolves to the value for the **Intermediate Directory** property.

$(OutDir)	Path to the output file directory, relative to the project directory. This path should have a trailing slash. This resolves to the value for the **Output Directory** property.
$(DevEnvDir)	The installation directory of Visual Studio 2010 (defined as drive + path); includes the trailing backslash '\'.
$(InputDir)	(Deprecated; migrated.) The directory of the input file (defined as drive + path); includes the trailing backslash '\'. If the project is the input, then this macro is equivalent to **$(ProjectDir)**.
$(InputPath)	(Deprecated; migrated.) The absolute path name of the input file (defined as drive + path + base name + file extension). If the project is the input, then this macro is equivalent to **$(ProjectPath)**.
$(InputName)	(Deprecated; migrated.) The base name of the input file. If the project is the input, then this macro is equivalent to **$(ProjectName)**.
$(InputFileName)	(Deprecated; migrated.) The file name of the input file (defined as base name + file extension). If the project is the input, then this macro is equivalent to **$(ProjectFileName)**.
$(InputExt)	(Deprecated; migrated.) The file extension of the input file. It includes the '.' before the file extension. If the project is the input, then this macro is equivalent to **$(ProjectExt)**.
$(ProjectDir)	The directory of the project (defined as drive + path); includes the trailing backslash '\'.
$(ProjectPath)	The absolute path name of the project (defined as drive + path + base name + file extension).
$(ProjectName)	The base name of the project.
$(ProjectFileName)	The file name of the project (defined as base name + file extension).
$(ProjectExt)	The file extension of the project. It includes the '.' before the file extension.
$(SolutionDir)	The directory of the solution (defined as drive + path); includes the trailing backslash '\'.
$(SolutionPa	The absolute path name of the solution (defined as drive + path + base name + file extension).

th)	
$(SolutionName)	The base name of the solution.
$(SolutionFileName)	The file name of the solution (defined as base name + file extension).
$(SolutionExt)	The file extension of the solution. It includes the '.' before the file extension.
$(TargetDir)	The directory of the primary output file for the build (defined as drive + path); includes the trailing backslash '\'.
$(TargetPath)	The absolute path name of the primary output file for the build (defined as drive + path + base name + file extension).
$(TargetName)	The base name of the primary output file for the build.
$(TargetFileName)	The file name of the primary output file for the build (defined as base name + file extension).
$(TargetExt)	The file extension of the primary output file for the build. It includes the '.' before the file extension.
$(VSInstallDir)	The directory into which you installed Visual Studio 2010. This property contains the version of the targeted Visual Studio, which might be different that the host Visual Studio. For example, when building with `$(PlatformToolset) = v90`, **$(VSInstallDir)** contains the path to the Visual Studio 2008 installation.
$(VCInstallDir)	The directory into which you installed Visual C++ 2010. This property contains the version of the targeted Visual C++, which might be different that the host Visual Studio. For example, when building with `$(PlatformToolset) = v90`, **$(VCInstallDir)** contains the path to the Visual C++ 2008 installation.
$(FrameworkDir)	The directory into which the .NET Framework was installed.
$(FrameworkVersion)	The version of the .NET Framework used by Visual Studio. Combined with **$(FrameworkDir)**, the full path to the version of the .NET Framework use by Visual Studio.

$(FrameworkSDKDir)	The directory into which you installed the .NET Framework. The .NET Framework could have been installed as part of Visual Studio 2010 or separately.
$(WebDeployPath)	The relative path from the web deployment root to where the project outputs belong. Returns the same value as RelativePath.
$(WebDeployRoot)	The absolute path to the location of **<localhost>**. For example, c:\inetpub\wwwroot.
$(SafeParentName)	(Deprecated.) The name of the immediate parent in valid name format. For example, a form is the parent of a .resx file.
$(SafeInputName)	(Deprecated.) The name of the file as a valid class name, minus file extension.
$(SafeRootNamespace)	(Deprecated.) The namespace name in which the project wizards will add code. This namespace name will only contain characters that would be permitted in a valid C++ identifier.
$(FxCopDir)	The path to the fxcop.cmd file. The fxcop.cmd file is not installed with all Visual C++ editions.

See Also

Other Resources

Building C++ Projects in Visual Studio