**DXCore – Contexts overview**

September 25th, 2010

The **context** determines if a particular feature is appropriate for use when working inside Visual Studio IDE. It is useful for distinguishing conditions where a feature should be available. If a feature with a specified **context** is available, then it means that the **context** is satisfied. For example, you may apply a **context** for a feature to be available while editing a source code, but not while IDE is in design-time or a model dialog is open.

A single **context** is provided by the [DXCore ContextProvider component](http://skorkin.com/2010/09/dxcore-components-contextprovider/" \o "DXCore Components - ContextProvider" \t "_blank). Registered **contexts** are available to users through the visual [ContextPicker component](http://skorkin.com/2010/09/dxcore-components-contextpicker/" \o "DXCore Components - ContextPicker" \t "_blank) of [DXCore](http://skorkin.com/2010/06/devexpress-dxcore-framework/" \o "DevExpress DXCore framework" \t "_blank). Using **ContextPicker** you can bind a particular context to a shortcut, code template and other [CodeRush](http://skorkin.com/2010/06/devexpress-coderush-pro/" \o "DevExpress CodeRush Pro" \t "_blank) solutions in the [Options Dialog](http://skorkin.com/2010/08/ide-tools-options-dialog/). **Contexts** can also be [verified programmatically](http://skorkin.com/2010/09/how-to-programmatically-verify-a-context/) inside your [DXCore plug-in](http://skorkin.com/2010/08/dxcore-plug-ins-overview/" \o "DXCore plug-ins overview" \t "_blank).

Here is a table of **contexts** shipped with the **DXCore** installation (total: 182). They are split into the following sections:

* *Editor\Clipboard\*
* *Editor\Code\Analysis\*
* *Editor\Code\*
* *Editor\*
* *Editor\Line\*
* *Editor\Selection\*
* *Editor\XML\*
* *File\*
* *Focus\*
* *Focus\Tool Windows\*
* *Language\*
* *Project\*
* *System\*
* **Editor\Clipboard\**

|  |  |
| --- | --- |
| **Context name** | **Description** |
| EnumerableTypeIsObject | Satisfied if the enumerable type on the clipboard has object type. |
| HasArrayType | Satisfied if the clipboard holds an identifier of the array type. |
| HasBoolean | Satisfied if the clipboard holds an identifier of type System.Boolean. |
| HasEnum | Satisfied if the clipboard holds an enum element. |
| HasIdentifier | Satisfied if the clipboard holds an identifier. |
| HasIdentifierWithMethod | Satisfied if the clipboard holds an identifier with a method matching the specified name (passed as a parameter to this context). |
| HasIdentifierWithProperty | Satisfied if the clipboard holds an identifier with a property matching the specified name (passed as a parameter to this context). |
| HasImplicitVariableName | Satisfied if the clipboard holds a name of an implicit variable. |
| HasInteger | Satisfied if the clipboard holds an identifier of type integer (e.g., System.Int16, System.Int32, System.Int64, System.UInt16, etc.). |
| HasMethodName | Satisfied if the clipboard holds a name of a method. |
| HasMultipleLines | Satisfied if the clipboard has text that includes one or more line terminators (e.g., a carriage return and/or line feed). |
| HasString | Satisfied if the clipboard holds an identifier of type System.String. |
| HasType | Satisfied if the clipboard holds an identifier of the specified type (passed as a parameter to this context). |
| Implements | Satisfied if the identifier on the clipboard implements the specified interface (passed as a parameter to this context). The interface name specified must be the full name (e.g., System.Collections.IEnumerable). |

* **Editor\Code\Analysis\**

|  |  |
| --- | --- |
| **Context name** | **Description** |
| AssignmentExist | Satisfied if the assignment on the current line has the specified type (passed as parameter to this context). |
| ConstructorHasParameter | Satisfied if the constructor to the specified type full name (e.g., “System.Collections.Hashtable”) accepts parameters or has more than one overload. |
| DeclaresLocalOfType | Satisfied if a local variable or parameter in the active method or property matches (or descends from) the specified type. |
| Expecting Assignment | Satisfied when the editor caret is inside an assignment or inside the expression of a assignment or inside the initializer of value. |
| HasStaticInitializer | Satisfied if the specified type full name (e.g., “System.Drawing.Color”) is a value type and has a static property of the same type (e.g., Empty, MinValue, etc.). |
| HasStaticMember | Satisfied if the specified type full name (e.g., “System.Drawing.Color”) has a static member of the specified name (e.g., “Empty”). |
| IsBuiltInType | Satisfied if the specified type name (e.g., “System.Int32” or “int”) is a primitive type. |
| IsDisposable | Satisfied if the specified type (or an ancestor) implements IDisposable. |
| IsSealedType | Satisfied if the specified type is a sealed type. |
| IsValueType | Satisfied if the specified type is a value type. |
| TypeImplements | Satisfied if the specified type implements the specified interface (passed as parameters to this context). The interface name specified must be the full name (e.g., System.Collections.IEnumerable). |

* **Editor\Code\**

|  |  |
| --- | --- |
| **Context name** | **Description** |
| AssemblyReferenced | Satisfied if the project references the specified assembly. |
| BlockFollowsCaret | Satisfied if a code block (e.g., “{…}” in C#) follows the caret, separated only by white space and/or empty lines. |
| DeclaresLocal | Satisfied if the specified identifier (passed as a parameter to this context) is declared as a local or a parameter in the active property or method. |
| DeclaresMember | Satisfied if the specified identifier (passed as a parameter to this context) is immediately declared as a member in the active type. |
| HasMember | Satisfied if the specified identifier (passed as a parameter to this context) is declared as a member in the active type or an ancestor. |
| In XPO Persistent Class | Satisfied if the caret is inside a class that has a Persistent attribute and/or descends from XPObject, XPBaseObject, or XPCustomObject, and the class does not include a NonPersistent attribute. |
| InAbstractMember | Satisfied when the caret is inside an abstract member (method, property, event, constructor, field, etc.). |
| InArrayNameModifier | Satisfied when the editor caret is inside an array-size initializer. |
| InAttribute | Satisfied when the editor caret is inside an attribute. |
| InBreakableLoop | Satisfied when the editor caret is inside of a for, foreach, do or while loop. |
| InCase | Satisfied when the editor caret is inside a switch (or VB Select) case statement. |
| InClass | Satisfied when the editor caret is inside a class. |
| InCodeBlock | Satisfied when the editor caret is inside a method or property, but not inside a string or comment. |
| InComment | Satisfied when the editor caret is inside a comment. |
| InConstructor | Satisfied when the editor caret is inside a constructor. |
| InDelegateDeclaration | Satisfied when the editor caret is inside a delegate declaration. |
| InDelegateParameters | Satisfied when the editor caret is inside a delegate parameters. |
| InDestructor | Satisfied when the editor caret is inside a destructor. |
| InEnum | Satisfied when the editor caret is inside an enum declaration. |
| InEvent | Satisfied when the editor caret is inside an event. |
| InEventAccessor | Satisfied when the editor caret is inside an event add or remove accessor. |
| InExpression | Satisfied when the editor caret is inside an expression. |
| InFieldInitializer | Satisfied when the editor caret is inside the initializer of a field. |
| InheritsFrom | Satisfied if the active class descends from the specified type (passed as a parameter to this context). |
| InIdentifier | Satisfied when the caret is in an identifier that has been declared. |
| InInterface | Satisfied when the editor caret is inside an interface. |
| InManagedClass | Satisfied if the active class is a managed class. Useful in languages that allow both managed and unmanaged types to co-exist (like C++). |
| InManagedStruct | Satisfied if the active struct is a managed struct. Useful in languages that allow both managed and unmanaged types to co-exist (like C++). |
| InMethod | Satisfied when the editor caret is inside a method. |
| InMethodBlock | True if the caret is inside a method at a point where statements can be placed (e.g., inside a method but not inside the method declaration signature). For example, in C#, this would be between the start and end brace delimiters. |
| InModule | True if the caret is inside a Visual Basic Module. |
| InNativeClass | Satisfied if the active class is a native class. Useful in languages that allow both managed and unmanaged types to co-exist (like C++). |
| InNativeStruct | Satisfied if the active struct is a native struct. Useful in languages that allow both managed and unmanaged types to co-exist (like C++). |
| InParameter | Satisfied when the editor caret is inside a parameter to a method. |
| InParameterBlock | Satisfied when the editor caret is inside a method’s open and close parens. |
| InPreprocessorDirective | Satisfied when the editor caret is inside a preprocessor directive. |
| InProperty | Satisfied when the editor caret is inside a property. |
| InPropertyAccessor | Satisfied when the editor caret is inside a property get or set accessor. |
| InRefType | Satisfied if the active type is a managed type. Useful in languages that allow both managed and unmanaged types to co-exist (like C++). |
| Inside\Member | Satisfied when the editor caret is inside a member with the specified name. |
| Inside\Namespace | Satisfied when the editor caret is inside a namespace with the specified name. |
| Inside\Type | Satisfied when the editor caret is inside a type with the specified name. |
| InStaticClass | Satisfied when the caret is inside a static class, in VS 2005 and up. |
| InStaticMember | Satisfied when the caret is inside a static member (method, property, event, constructor, field, etc.). |
| InStaticType | Satisfied when the caret is inside a static type, in VS 2005 and up. |
| InString | Satisfied when the editor caret is inside a string. |
| InStruct | Satisfied when the editor caret is inside a struct. |
| InStyleCss | Satisfied when the editor caret is inside an html tag <style > </style>. |
| InSwitch | Satisfied when the editor caret is inside a switch (or VB Select) statement. |
| InType | Satisfied when the editor caret is inside a class, struct, or interface. |
| InUsingExpression | Satisfied when the editor caret is inside an expression of a using statement. |
| InValueClass | Satisfied if the active class is an unmanaged class. Useful in languages that allow both managed and unmanaged types to co-exist (like C++). |
| InValueStruct | Satisfied if the active struct is an unmanaged struct. Useful in languages that allow both managed and unmanaged types to co-exist (like C++). |
| InValueType | Satisfied if the active type is an unmanaged type. Useful in languages that allow both managed and unmanaged types to co-exist (like C++). |
| InXmlDocComment | Satisfied when the editor caret is inside an XML documentation comment. |
| InXMLLiteral | True if the caret is inside a XML literal. |
| Is Declared\Identifier | Satisfied if the specified identifier (passed as a parameter to this context) is declared. |
| Is Declared\Method | Satisfied when the specified method (passed as a parameter to this context) is declared. |
| Is Declared\Property | Satisfied when the specified property (passed as a parameter to this context) is declared. |
| IsNamespaceReferenced | Satisfied if the identifier on the clipboard implements the specified interface (passed as a parameter to this context). The interface name specified must be the full name (e.g., System.Collections.IEnumerable). |
| IsRefType | Satisfied if the active type is a managed type. Useful in languages that allow both managed and unmanaged types to co-exist (like C++). |
| Reflection\TypesMatch | Satisfied if the two types (passed as parameters to this context) match. Simple types are promoted to full type names before the comparison is performed (e.g., “bool” will match “System.Boolean” in C#). |

* **Editor\**

|  |  |
| --- | --- |
| **Context name** | **Description** |
| In Code Snippet Field | Satisfied if the editor caret is inside of a Visual Studio code snippet field. |
| Templates\First Character is Uppercase | Satisfied if the template at the caret starts from an uppercase character. |

* **Editor\Line\**

|  |  |
| --- | --- |
| **Context name** | **Description** |
| After First Char | Satisfied when the editor caret is to the right of the first non-white-space character on the line. |
| AppearsToDeclare | Satisfied if the specified identifier is partially declared on the active line. |
| At Compiler Directive | Satisfied when the caret is at the line with a compiler directive (e.g. #warning, #error). |
| At Region End | Satisfied when the editor caret is immediately to the left of an #endregion keyword. |
| At Region Start | Satisfied when the editor caret is immediately to the left of a #region keyword. |
| AtEndOfLine | Satisfied when the caret is at the end of a line. Selected text is, by default, ignored when determining whether the caret is at the end of a line or not. So if you have a selection starting at the end of a line and running to the middle of the line, the default behavior for this context is to be satisfied. You can change this behavior by passing false as the TreatSelectionAsEmptyText argument. |
| Before First Char | Satisfied when the caret is to the left of the first non-white-space character on the line. |
| Caret in Whitespace | Satisfied when the editor caret is surrounded by whitespace (e.g., space, tab, or line break chars).  Note: This context will return false if the caret is inside virtual space (e.g., after the last character of line or after the last line of a file). |
| Char Left Is | Satisfied when the character to the left of the editor caret matches the specified character (passed as a parameter to this context). |
| Char Right Is | Satisfied if the character to the right of the caret is the same as the specified character (passed as a parameter to this context). |
| HasLineContinuation | True if previous line has Visual Basic line-continuation character. |
| In Virtual Space | Satisfied when the editor caret is inside virtual space. |
| InParens | Satisfied when the editor caret is inside parentheses. |
| Left Text Contains | Satisfied when the text to the left of the caret contains the specified text (passed as a parameter to this context). |
| Left Text Matches RegEx | Satisfied if the text to the left of the caret matches the specified regular expression. |
| Needs Indenting | Satisfied if the caret is at the beginning of the code on a line (at the first token, with white space on the left or at column 1), and the line above is a parent to a single statement (like an if, for, else, while, etc.) and the code on the line above starts at the same display column as the code at the caret. |
| On First Line | Satisfied if the caret is on the first line of a file. |
| On Last Line | Satisfied if the caret is on the last line of a file. |
| OnEmptyLine | Satisfied when the editor caret is on an empty line (or on a line that contains only whitespace). When this context is checked before a template expands, the line is evaluated without the template text. Selected text is, by default, ignored when determining whether the caret is on an empty line or not. You can change this behavior by passing false as the TreatSelectionAsEmptyText argument. |
| Right Text Contains | Satisfied when the text to the right of the caret contains the specified text (passed as a parameter to this context). |
| Right Text Matches RegEx | Satisfied when the text to the right of the caret matches the specified regular expression (passed as a parameter to this context). |

* **Editor\Selection\**

|  |  |
| --- | --- |
| **Context name** | **Description** |
| Any Selection\Line Fragment | Satisfied when the current selection in the editor extends a portion of a single line of code. |
| Any Selection\Multiple Lines | Satisfied when the current selection in the editor extends across two or more lines. |
| Any Selection\Whole Line | Satisfied when the current selection in the editor encompasses a whole line. A whole line is defined as a selection that starts at the beginning of the line, and extends to the end of the same line or extends to just before the first character of the next line. |
| Equals | Satisfied if the text of the selection equals the specified text. |
| Identifier Name | Satisfied when the current selection in the editor extends the name it identifies. |
| No Selection | Satisfied when there is no selection in the editor. |

* **Editor\XML\**

|  |  |
| --- | --- |
| **Context name** | **Description** |
| InSection | Satisfied if the active XML node is parented (directly or indirectly) by the specified element name (passed as a parameter to this context). |
| NAnt\InProject | Satisfied when the active file is a Nant project and the caret is inside a project |
| NAnt\InTarget | Satisfied when the active file is a Nant project and the caret is inside a target. |
| NAnt\InTask | Satisfied when the active file is a Nant project and the caret is inside a specified task. |
| NAnt\InTaskAttribute | Satisfied when active file is a Nant project and caret is inside a specified attribute of specified task. |
| NAnt\IsNAntProjectFile | Satisfied when the active file is a Nant project. |
| ParentIs | Satisfied if the active XML node is directly parented by the specified element name (passed as a parameter to this context). |

* **File\**

|  |  |
| --- | --- |
| **Context name** | **Description** |
| IsWPF | Satisfied if the file references one or more WPF namespaces. |

* **Focus\**

|  |  |
| --- | --- |
| **Context name** | **Description** |
| Documents\HTML\HTML Designer | Satisfied when an HTML designer has focus. |
| Documents\HTML\HTML View | Satisfied when an HTML view has focus. |
| Documents\Source\Code Editor | Satisfied when the source code editor has focus. |
| Documents\Source\Form Designer | Satisfied when the Form Designer has focus. |
| Modal Dialog | Satisfied when a modal dialog has focus. |

* **Focus\Tool Windows\**

|  |  |
| --- | --- |
| **Context name** | **Description** |
| Autos | Satisfied when the Autos window has focus. |
| Debug Windows\Call Stack | Satisfied when the Call Stack has focus. |
| Debug Windows\Command | Satisfied when the Command window has focus. |
| Debug Windows\Locals | Satisfied when the Locals window has focus. |
| Debug Windows\Thread View | Satisfied when the Thread View has focus. |
| Debug Windows\Watch | Satisfied when the Watch window has focus. |
| Document Outline | Satisfied when the Document Outline window has focus. |
| Find Results | Satisfied when the Find Results window has focus. |
| Other ToolWindow | Satisfied when an unidentified tool window has focus. |
| Output | Satisfied when the Output window has focus. |
| Property Inspector | Satisfied when the Property Browser has focus. |
| Running Documents | Satisfied when the Running Documents window has focus. |
| Solution Explorer | Satisfied when the Solution Explorer has focus. |
| Task List | Satisfied when the Task List window has focus. |
| Tool Box | Satisfied when the Toolbox window has focus. |

* **Language\**

|  |  |
| --- | --- |
| **Context name** | **Description** |
| Active Language Supported | Satisfied if the active file is in a language supported by the DXCore. |
| C# | Satisfied if a C# file is active in the editor. |
| C++ | Satisfied if a C++ file is active in the editor. |
| IsKeyword | Satisfied when the string passed to it is keyword. |
| Matches | Satisfied when the language ID of the active file matches the specified ID (passed as a parameter to this context). |
| Visual Basic | Satisfied if a Visual Basic file is active in the editor. |
| XML or HTML | Satisfied when an XML or HTML file is active in the editor. |

* **Project\**

|  |  |
| --- | --- |
| **Context name** | **Description** |
| IsManaged | Satisfied if a project has managed compile options. |
| IsNative | Satisfied if project has native compile options. |
| IsWPF | Satisfied when the project references any of the PresentationCore, WindowsBase, or PresentationFramework assemblies. |
| References | Satisfied if the project references the specified assembly. |

* **System\**

|  |  |
| --- | --- |
| **Context name** | **Description** |
| Can Activate Next Field | Satisfied if two or more Fields exist and a Field is active. You can add fields to a text expansion using the Field TextCommand. |
| Code Issues Zoom Window is active | Satisfied if the Code Issues Zoom Window hint is active. |
| CodeFix UI is active | Satisfied if the CodeFix hint is active. |
| Debug Mode | Satisfied if Visual Studio is in debug mode. |
| Design Mode | Satisfied if Visual Studio is in design mode. |
| IME Composition Active | Satisfied when an IME (input method editor) is in an active composition state. IMEs are typically used to enter Unicode characters. |
| In Linked Identifier | Satisfied if a linked identifier is active. |
| In Nav Field | Satisfied if a navigation field is active. |
| In Nav Link | Satisfied if a navigation identifier is active. |
| In TextField | Satisfied if a Field is active. You can add fields to a text expansion using the Field TextCommand. |
| Incremental Search Active | Satisfied if the Incremental Search is active in the editor. |
| Intellassist Active | Satisfied when the [CodeRush Intellassist](http://www.skorkin.com/2011/10/coderush-code-editor-auto-complete-features-intellassist/" \o "Coderush code editor auto-complete features – Intellassist) feature is active. |
| Intellassist Suspended | Determines whether or not Intellassist is suspended. |
| Intellisense Active | Satisfied when Visual Studio’s Intellisense is active. |
| Menu Active | Satisfied when any menu is active in Visual Studio. |
| Picker Is Active | Satisfied when the Refactor UI is active. |
| RefactorInstalled | Satisfied when the Refactor! Pro is installed. |
| Text Replacer Is Active | Satisfied when text replacer is active. |
| Tip Window Active | Satisfied when a Visual Studio tool tip window is active. |
| VariableMatches | Satisfied if the specified template variable matches the specified value. |
| Visual Studio SmartTag Active | Satisfied when Visual Studio’s smarttag visible and has expanded window. |
| VSVersion\VS2002 | Satisfied if the DXCore is running in Visual Studio .NET 2002. |
| VSVersion\VS2003 | Satisfied if the DXCore is running in Visual Studio .NET 2003. |
| VSVersion\VS2005 | Satisfied if the DXCore is running in Visual Studio 2005. |
| VSVersion\VS2008 | Satisfied if the DXCore is running in Visual Studio 2008. |
| VSVersion\VS2010 | Satisfied if the DXCore is running in Visual Studio 2010. |
| Waiting for second key of chord | Satisfied if Visual Studio is waiting for the second key of a shortcut chord. |

These are additional contexts that ship with the [CodeRush Pro](http://skorkin.com/2010/06/devexpress-coderush-pro/" \o "DevExpress CodeRush Pro" \t "_blank) product (total: 14):

* **Editor\**

|  |  |
| --- | --- |
| **Context name** | **Description** |
| Code\ColorIsActive | Satisfied if the caret is inside an editable color declaration. |
| Selection\Lines End With | Satisfied if each non-empty line in the selection starts and ends with the specified text. |
| Selection\Lines Start With | Satisfied if each non-empty line in the selection starts with the specified text. |
| Selection\Lines Wrapped With | Satisfied if each non-empty line in the selection starts and ends with the specified text. |
| Selection\ExpressionFocus Available | Satisfied if it’s appropriate to use the Tab and/or Shift+Tab keys to set focus to the next or previous expression. In general, this means there is no selection and the caret is not adjacent to leading or trailing white space (where the Tab key would be useful for indenting the code), or it means a selection exists that contains a precise expression (without including leading or trailing white space). |
| Selection\Cell Navigation Available | Satisfied if it’s appropriate to use the Tab and/or Shift+Tab keys to set focus to the next or previous cell in a table. In general, this means there is no selection and the caret is either on an empty line or at least not inside a leading white space (e.g., where the Tab key might be useful for indenting), or it means a selection exists that precisely contains the contents of a cell. |

* **System\**

|  |  |
| --- | --- |
| **Context name** | **Description** |
| CellNavTooltipIsActive | Satisfied if the Cell Nav tooltip is active. |
| QuickNav Active | Satisfied if the QuickNav or QuickFileNav windows are active. |
| References Window is active | Satisfied when the references window is active. |
| Selection Inversion Available | Satisfied when the [Selection Inversion](http://www.skorkin.com/2011/10/inverting-a-code-block-into-its-opposite-using-coderush-selection-inversion/) feature is available. |

* **System\AutoCompletion\**

|  |  |
| --- | --- |
| **Context name** | **Description** |
| Inserted Closing Bracket on this Line | Satisfied if the Smart Expressions plug-in has recently inserted a closing bracket on this line. |
| Inserted Closing Paren on this Line | Satisfied if the Smart Expressions plug-in has recently inserted a closing parenthesis on this line. |
| Just Inserted Closing Bracket | Satisfied if the Smart Expressions plug-in has just inserted a closing bracket. |
| Just Inserted Closing Paren | Satisfied if the Smart Expressions plug-in has just inserted a closing parenthesis. |

*—–*