**FxCop Naming Conventions and Guidelines**

**Avoid language specific type names in parameters**

TypeName: AvoidLanguageSpecificTypeNamesInParameters

Category: Microsoft.Naming

Message Level: Error

Each concatenated word in the parameter name is checked against the following language-specific type names, in a case-insensitive manner. Language-specific type names might not be intuitive for developers using other languages.

Ex: bool , int , float32

**Avoid type names in parameters**

TypeName: AvoidTypeNamesInParameters

Category: Microsoft.Naming

Message Level: Error

Parameter names are better used to convey a parameter's meaning than to describe a parameter's type, which is expected to be provided by development tools.

Ex: valBool, valFloat

**Compound words should be cased correctly**

TypeName: CompoundWordsShouldBeCasedCorrectly

Category: Microsoft.Naming

Message Level: Error

The name of the identifier is split into words based on the casing. Each contiguous two word combination is checked by the Microsoft spelling checker library. If recognized, the identifier produces a violation of the rule.

Ex: countOfObjects, totalItems

**Do not name enum values 'Reserved'**

TypeName: DoNotNameEnumValuesReserved

Category: Microsoft.Naming

Message Level: Error

This rule assumes that an enumeration member with a name that contains "reserved" is not currently used but is a placeholder to be renamed or removed in a future version. Instead of using a reserved member, add a new member to the enumeration in the future version. In most cases, the addition of the new member is not a breaking change as long as the addition does not cause the values of the original members to change.

**Do not prefix enum values with type name**

TypeName: DoNotPrefixEnumValuesWithTypeName

Category: Microsoft.Naming

Message Level: Error

Names of enumeration members are not prefixed with the type name because type information is expected to be provided by development tools.

Ex:

public enum DigitalImageMode

{

DigitalImageModeBitmap = 0,

DigitalImageModeGrayscale = 1,

DigitalImageModeRGB = 2

}

**Events should not have before or after prefix**

TypeName: EventsShouldNotHaveBeforeOrAfterPrefix

Category: Microsoft.Naming

Message Level: Error

Event names should describe the action that raises the event. To name related events that are raised in a specific sequence, use the present or past tense to indicate the relative position in the sequence of actions.

Ex: Closing and Closed, instead of BeforeClose and AfterClose

**Flags enums should have plural names**

TypeName: FlagsEnumsShouldHavePluralNames

Category: Microsoft.Naming

Message Level: Error

Types marked with FlagsAttribute have names that are plural because the attribute indicates that more than one value can be specified.

Ex: Days instead of Days

**Identifiers should be cased correctly**

TypeName: IdentifiersShouldBeCasedCorrectly

Category: Microsoft.Naming

Message Level: Error

By convention, parameter names use camel casing; **namespace**, **type**, and **member** names use Pascal casing. In a **camel-cased** name, the first letter is lower case, and the first letter of any remaining words in the name is in uppercase. In a **Pascal-cased** name, the first letter is upper case, and the first letter of any remaining words in the name is in uppercase.

Ex: packetSniffer, ioFile (camel cased)

PacketSniffer, IOFile (Pascal cased)

**Identifiers should be spelled correctly**

TypeName: IdentifiersShouldBeSpelledCorrectly

Category: Microsoft.Naming

Message Level: Error

This rule parses (breaks into multiple words up to the last capital letter) the identifier into tokens and checks the spelling of each token.

**Identifiers should differ by more than case**

TypeName: IdentifiersShouldDifferByMoreThanCase

Category: Microsoft.Naming

Message Level: Error

Ex: Id, ID, iD - Invalid

**Identifiers should have correct prefix**

TypeName: IdentifiersShouldHaveCorrectPrefix

Category: Microsoft.Naming

Message Level: Error

By convention, the names of certain programming elements start with a specific prefix.

Ex: I for interfaces

**Identifiers should have correct suffix**

TypeName: IdentifiersShouldHaveCorrectSuffix

Category: Microsoft.Naming

Message Level: Error

By convention, the names of types that extend certain base types or that implement certain interfaces, or types derived from these types, have a suffix that is associated with the base type or interface.

Ex: QueueCollection, QueueDictionary

Helps in deciding whether to use,

foreach(SomeType x in SomeCollection) { } (for collections)

Or,

foreach(SomeType x in SomeDictionary.Values) { } (for dictionaries)

Below table gives a summary or suffixes:

|  |  |
| --- | --- |
| **Base type/Interface** | **Suffix** |
| [System.Attribute](http://msdn.microsoft.com/library/en-us/cpref/html/frlrfsystemattributeclasstopic.asp) | Attribute |
| [System.EventArgs](http://msdn.microsoft.com/library/en-us/cpref/html/frlrfsystemeventargsclasstopic.asp) | EventArgs |
| [System.Exception](http://msdn.microsoft.com/library/en-us/cpref/html/frlrfsystemexceptionClasstopic.asp) | Exception |
| [System.Collections.ICollection](http://msdn.microsoft.com/library/en-us/cpref/html/frlrfsystemCollectionsICollectionClasstopic.asp) | Collection |
| [System.Collections.IDictionary](http://msdn.microsoft.com/library/en-us/cpref/html/frlrfsystemCollectionsIDictionaryClasstopic.asp) | Dictionary |
| [System.Collections.IEnumerable](http://msdn.microsoft.com/library/en-us/cpref/html/frlrfsystemCollectionsIEnumerableClasstopic.asp) | Collection |
| [System.Collections.Queue](http://msdn.microsoft.com/library/en-us/cpref/html/frlrfsystemcollectionsqueueclasstopic.asp) | Collection or Queue |
| [System.Collections.Stack](http://msdn.microsoft.com/library/en-us/cpref/html/frlrfsystemcollectionsStackclasstopic.asp) | Collection or Stack |
| [System.Collections.Generic.ICollection](mk:@MSITStore:C:\Program%20Files\Microsoft%20FxCop%201.35\FxCop.chm::/Docs/Rules/Naming/System.Collections.Generic.ICollection) | Collection |
| [System.Collections.Generic.IDictionary](mk:@MSITStore:C:\Program%20Files\Microsoft%20FxCop%201.35\FxCop.chm::/Docs/Rules/Naming/System.Collections.Generic.IDictionary) | Dictionary |
| [System.Data.DataSet](http://msdn.microsoft.com/library/en-us/cpref/html/frlrfSystemDataDataSetClassTopic.asp) | DataSet |
| [System.Data.DataTable](http://msdn.microsoft.com/library/en-us/cpref/html/frlrfSystemDataDataTableClassTopic.asp) | Collection or DataTable |
| [System.IO.Stream](http://msdn.microsoft.com/library/en-us/cpref/html/frlrfSystemIOStreamClassTopic.asp) | Stream |
| [System.Security.IPermission](http://msdn.microsoft.com/library/en-us/cpref/html/frlrfSystemSecurityIPermissionClassTopic.asp) | Permission |
| [System.Security.Policy.IMembershipCondition](http://msdn.microsoft.com/library/en-us/cpref/html/frlrf%20SystemSecurityPolicyIMembershipConditionClassTopic.asp) | Condition |
| An event-handler delegate. | EventHandler |

**Identifiers should not contain underscores**

TypeName: IdentifiersShouldNotContainUnderscores

Category: Microsoft.Naming

Message Level: Error

**Identifiers should not have incorrect prefix**

TypeName: IdentifiersShouldNotHaveIncorrectPrefix

Category: Microsoft.Naming

Message Level: Error

By convention, only certain programming elements have names that begin with a specific prefix. Type names do not have a specific prefix and should not be prefixed with a 'C'.

**Identifiers should not have incorrect suffix**

TypeName: IdentifiersShouldNotHaveIncorrectSuffix

Category: Microsoft.Naming

Message Level: Error

By convention, only the names of types that extend certain base types or that implement certain interfaces, or types derived from these types, should end with specific reserved suffixes. Other type names should not use these reserved suffixes. In addition, the following suffixes should not be used:

Delegate

Enum

Flags for an enumeration

Impl

**Identifiers should not match keywords**

TypeName: IdentifiersShouldNotMatchKeywords

Category: Microsoft.Naming

Message Level: Error

Identifiers for namespaces and types should not match keywords defined by languages that target the common language runtime. Depending on the language in use and the keyword, compiler errors and ambiguities can make the library difficult to use.

**Long acronyms should be pascal-cased**

TypeName: LongAcronymsShouldBePascalCased

Category: Microsoft.Naming

Message Level: Error

Ex: DB, CR, Cpa and Ecma - valid

Io, XML and DoD - invalid

**Only FlagsAttribute enums should have plural names**

TypeName: OnlyFlagsEnumsShouldHavePluralNames

Category: Microsoft.Naming

Message Level: Error

Naming conventions dictate that a plural name for an enumeration indicates that more than one value of the enumeration can be specified simultaneously. The FlagsAttribute tells compilers that the enumeration should be treated as a bit field that allows bitwise operations on the enumeration.

**Parameter names should match base declaration**

TypeName: ParameterNamesShouldMatchBaseDeclaration

Category: Microsoft.Naming

Message Level: Error

Consistent naming of parameters in an override hierarchy increases the usability of the method overrides. A parameter name in a derived method that differs from the name in the base declaration can cause confusion as to whether the method is an override of the base method or a new overload of the method.

**Parameter names should not match member names**

TypeName: ParameterNamesShouldNotMatchMemberNames

Category: Microsoft.Naming

Message Level: Error

A parameter name should convey a parameter's meaning and a member name should convey a member's meaning. It would be a rare design where these were the same. Naming a parameter the same as its member name is unintuitive and makes the library difficult to use.

**Property names should not match get methods**

TypeName: PropertyNamesShouldNotMatchGetMethods

Category: Microsoft.Naming

Message Level: Error

Get methods and properties should have names that clearly distinguish their function.

Ex: Date(property)

GetDate(method)

**Resource string compound words should be cased correctly**

TypeName: ResourceStringCompoundWordsShouldBeCasedCorrectly

Category: Microsoft.Naming

Message Level: Error

Each word in the resource string is split into tokens based on the casing. Each contiguous two token combination is checked by the Microsoft spelling checker library. If recognized, the word produces a violation of the rule.

Ex: CheckSum, MultiPart - invalid

Checksum and Multipart - valid

**Resource strings should be spelled correctly**

TypeName: ResourceStringsShouldBeSpelledCorrectly

Category: Microsoft.Naming

Message Level: Error

This rule parses the resource string into words (tokenizing compound words) and checks the spelling of each word/token.

**Short acronyms should be uppercase**

TypeName: ShortAcronymsShouldBeUppercase

Category: Microsoft.Naming

Message Level: Error

By convention, two-letter acronyms use all uppercase letters, and acronyms of three or more characters use Pascal casing.

'ID' is special-cased to cause a violation of this rule because 'Id' is not an acronym but is an abbreviation for identification.

**Type names should not match namespaces**

TypeName: TypeNamesShouldNotMatchNamespaces

Category: Microsoft.Naming

Message Level: Error

Type names should not match the names of namespaces defined in the .NET Framework class library. Violating this rule can reduce the usability of the library.

**Use preferred terms**

TypeName: UsePreferredTerms

Category: Microsoft.Naming

Message Level: Error

This rule parses an identifier into tokens. Each single token and each contiguous dual token combination is compared against terms built into the rule and in the Deprecated section of any custom dictionaries.

A table summarizing the obsolete terms and their replacement:

|  |  |
| --- | --- |
| **Obsolete term** | **Preferred term** |
| ComPlus | EnterpriseServices |
| Cancelled | Canceled |
| Indices | Indexes |
| LogIn | LogOn |
| LogOut | LogOff |
| SignOn | SignIn |
| SignOff | SignOut |
| Writeable | Writable |