Mappings between C# and UML elements

In class diagrams, Unified Modeling Language (UML) notation is used to represent C# and CTS elements, the fields and methods that comprise an element, and the relationships between C# elements.

Class diagrams represent not only C# classes, but other C# elements such as structs, enums, and delegates.

A stereotype is an extension mechanism that broadens the vocabulary of the UML and gives more specific meaning to a C# class and other elements.

The following table summarizes the mapping between C# elements and UML elements in class diagrams.

Table 1.

.NET elements	UML elements
File	Artifact with a «C# File» or «.NET Assembly File» stereotype.
Folder	Artifact with a «C# Folder» stereotype.
Namespace	Package with a «C# Namespace» stereotype and the namespace icon.
Class	Class with the «C# Class» stereotype.
Enum	Enumeration with or without an «C# enumeration» stereotype.
Struct	Class with «C# Struct» stereotype and class icon.
Interface	Interface with the interface icon.
Partial class	Class with a «C# Partial Class» stereotype.
Partial interface	Interface with a «C# Partial Interface» stereotype.
Delegate	Class with a «C# Delegate» stereotype. The invocation method and its signature are mapped using a UML operation within this class. The name of this operation is the same as the class. Its signature is the signature of the delegate. By default, this delegate class will derive from System.Delegate or System.MultiCastDelegate.
Member field	Property. Depending on user preferences, an association for non-primitive types also is created. The diagrams support showing properties as associations without the association needing to be explicitly created. The Eclipse icon or the UML visibility icon will be shown depending upon the user's preference.
Event	This is modeled like a member field, but with an «C# Event» stereotype.
Indexer	This is modeled like a member field, but with an «C# Indexer» stereotype.
Property	This is modeled like a member field, but with a «C# Property» stereotype. Each property has accessors that can be explicitly modeled as operations with the «get» and «set» stereotypes.
Methods	Operation. The Eclipse icon or the UML visibility icon is shown depending upon the user's preference. **Note:** C# static methods can be visualized, but cannot be added to diagrams.

• Value parameters • Reference parameters, declared with ref modifier • Output parameters, declared with out modifier • Parameter arrays, declared with the params modifier	• Parameter with direction [in]. • Parameter with direction [inout]. • Parameter with direction [out]. • Parameter with multiplicity set to "*" to indicate a single dimension array.
Attributes	Class with an «C# Attribute» stereotype. This comes with a property, "AttributeUsage," that should be set to the usage string. Attributes are specified in free form text within the documentation view of the corresponding UML element of each attribute target. The complete string within the square brackets "[]" is specified in the documentation view. Valid attribute targets are assembly, event, field, method, param, property, returntype, typevar.
Generics	Generics map to UML template parameters. Classes, structs, interfaces and methods can be parameterized. These map to parameterized UML classes, parameterized UML classes with a «struct» stereotype, parameterized UML interfaces and parameterized UML operations, respectively.
Nullable types	Nullable types for a predefined type map to UML primitive types with the same name. Nullable types for other value types are mapped to a UML class with the «nullable» stereotype applied, named as the base value type and a UML substitution relationship existing from the UML lass to the UML type representing the base value type.
Accessibility (public, private, protected, internal, protected internal)	For classes, attributes, and operations the accessibility maps to a visibility or keyword property of the same name.

Note: The name of the stereotype is based on whether the element is a C# element or a .NET Assembly element. The stereotypes in the table above specify only the C# form. For example, a visualized namespace can have the following stereotypes:

- <<.NET Assembly Namespace>>

  The namespace is from the .NET assembly.

- <<C# Namespace>>

  The namespace is from a C# file.