Extern alias directive

Until now, C# has supported only a single namespace hierarchy into which types from referenced assemblies and the current program are placed. Because of this design, it has not been possible to reference types with the same fully qualified name from different assemblies, a situation that arises when types are independently given the same name, or when a program needs to reference several versions of the same assembly. Extern aliases make it possible to create and reference separate namespace hierarchies in such situations.

An *extern-alias-directive* introduces an identifier that serves as an alias for a namespace hierarchy.

Consider the following two assemblies:

Assembly a1.dll:

```
namespace N
{
public class A {}
public class B {}
}
```

Assembly a2.dll:

```
namespace N
{
public class B {}
public class C {}
}
```

and the following program:

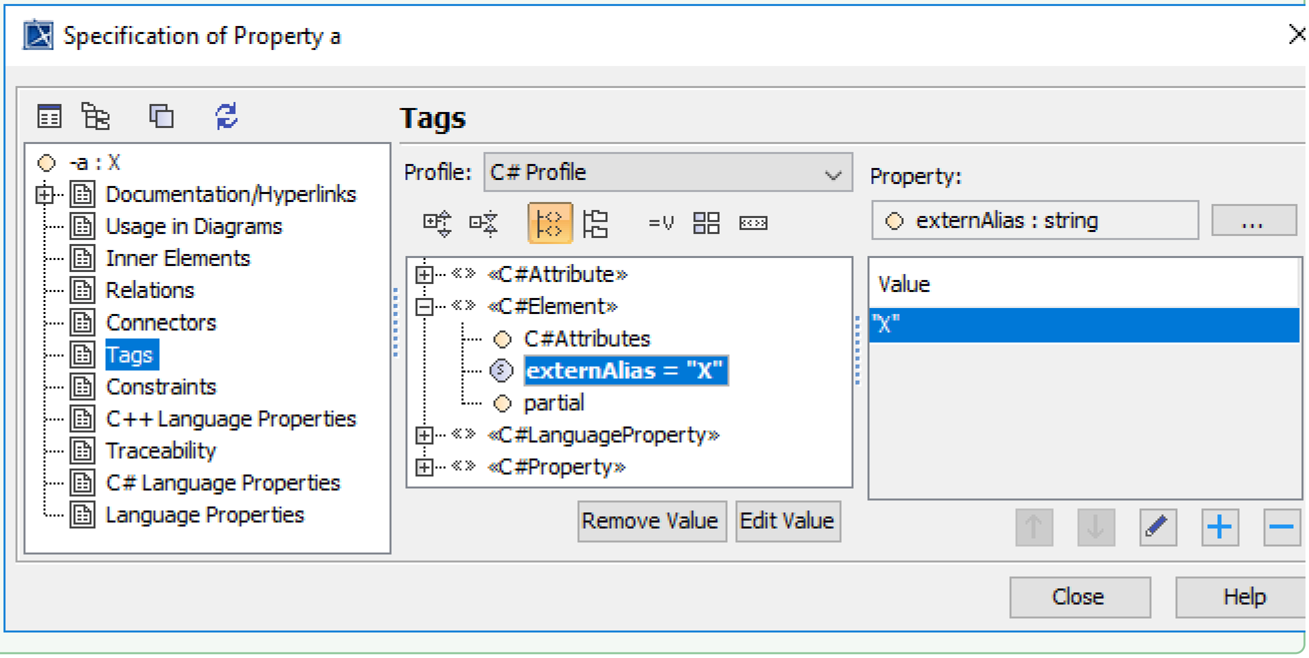
```
class Test

{

N.A a;

N.C c;

}
```

The following program declares and uses two extern aliases, X and Y, each of which represent the root of a distinct namespace hierarchy created from the types contained in one or more assemblies.

Code:

```
extern alias X;
extern alias Y;
class Test
{
        X::N.A a;
        X::N.B b1;
        Y::N.B b2;
        Y::N.C c;
}
```

Setting Tags for <<C#Element>>, externAlias:



Specification of Property a

Tags

Profile: C# Profile

Property: externAlias : string

externAlias = "X"

Value

"X"