Accessor declarations

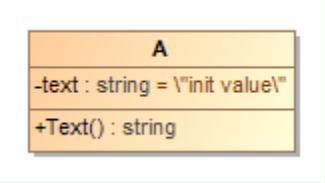
The use of *accessor-modifiers* is governed by the following restrictions:

- An *accessor-modifier* may not be used in an interface or in an explicit interface member implementation.
- For a property or indexer that has no override modifier, an *accessor-modifier* is permitted only if the property or indexer has both a get and set acc essor, and then is permitted only on one of those accessors.
- For a property or indexer that includes an override modifier, an accessor must match the *accessor-modifier*, if any, of the accessor being overridden.
- The *accessor-modifier* must declare an accessibility that is strictly more restrictive than the declared accessibility of the property or indexer itself. To be precise:
- If the property or indexer has a declared accessibility of public, any *accessor-modifier* may be used.
- If the property or indexer has a declared accessibility of protected internal, the *accessor-modifier* may be either internal, protected, or private.
- If the property or indexer has a declared accessibility of internal or protected, the *accessor-modifier* must be private.
- If the property or indexer has a declared accessibility of private, no *accessor-modifier* may be used.

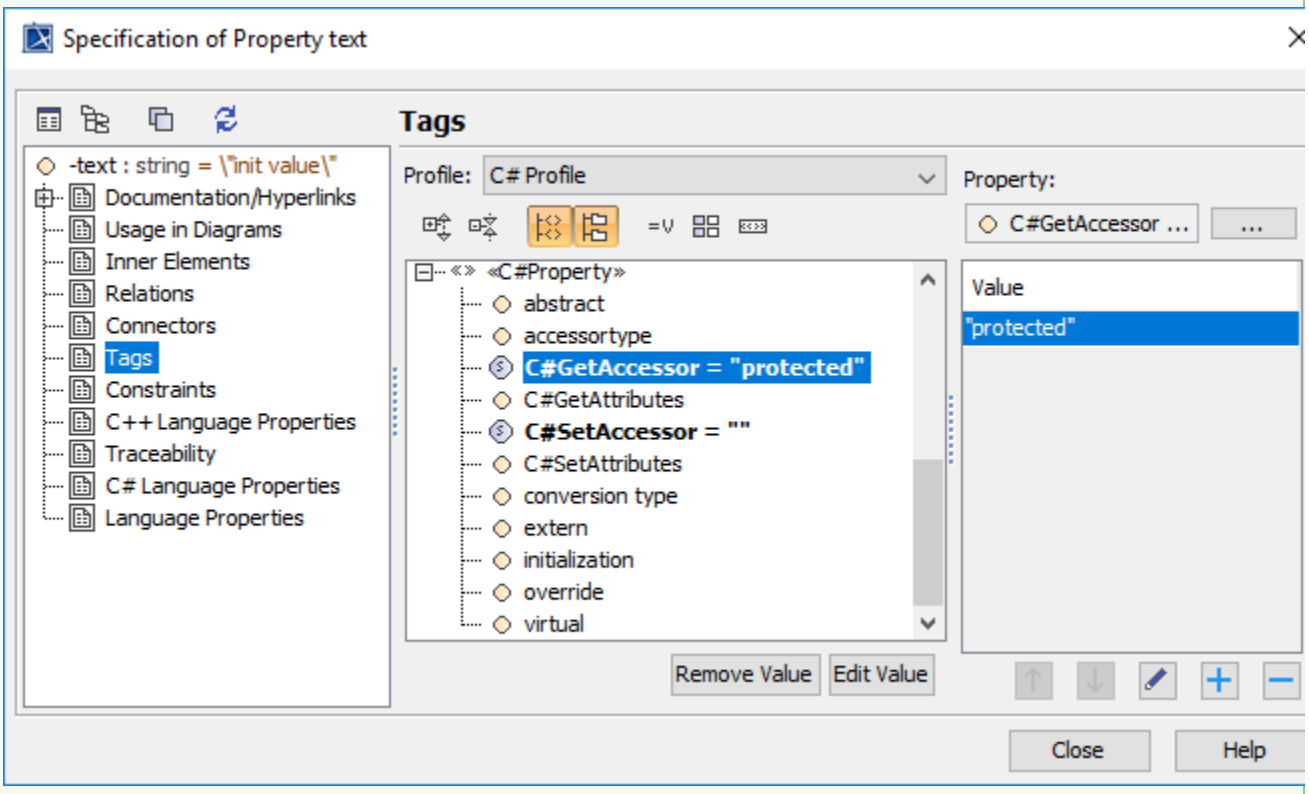
Example

```
Class A
{
private string text = “init value”;
public String Text
{
       protected get{ return text;}
       set{ text = value;}
}
}
```

Reversed UML model:



Example of Tags required when an accessor-modifier is permitted only if the property or indexer has both a get and set accessor, and then is permitted only on one of those accessors:



Property *text* has both a get and set accessor.