Hiding a function

Asked 16 years, 2 months ago Modified 5 years, 2 months ago

Viewed 465 times

2

I have a class holding complex scientific computations. It is set up to only allow a user to create a properly instantiated case. To properly test the code, however, requires setting internal state variables directly, since the reference documents supply this data in their test cases. Done improperly, however, it can invalidate the state.

So I must have the ability, a member function, to set internal variables from the unit test programs. But I want to **strongly discourage** normal users from calling this function. (Yes, a determined user can muck with anything... but I don't want to *advertise* that there is a way to do something *wrong*.)

It would be nice to be able to tell Intellisense to not show the function, for instance.

The best solution I have at the moment is to just name the function something like: DangerousSet().

What other options do I have?

Follow-Up

I found Amy B's answer most useful to my situation. Thanks!

Mufasa's suggestion to use reflection was great, but harder to implement (for me).
Chris' suggestion of using a decorator was good, but didn't pan out.
BFree's suggestion on XML is also good, and was already in use, but doesn't really solve the problem.

Finally, BillTheLizard's suggestion that the problem is in the source documents is not something I can control. International experts publish highly technical books and journal articles for use by their community. The fact that they don't address my particular needs is a fact of life. There simply are no alternative documents.

c# visual-studio intellisense

Share
Improve this question
Follow

edited Oct 4, 2019 at 22:48

Amy B
110k ● 21 ● 139 ● 190

asked Oct 3, 2008 at 16:30

Mark T
3,484 ● 5 ● 34 ● 45

7 Answers

Sorted by: Highest score (default)

14

You can use InternalsVisibleToAttribute to mark internal members as visible to your test assembly. It seems to shine when used in this context, though its not quite "friend".

1. Mark your `DangerousSet` function `internal` instead of `public`.
2. In Properties\AssemblyInfo.cs of the project containing `DangerousSet`:

   `[assembly:InternalsVisibleTo("YourTestAssembly")]`

If you have two test assemblies for whatever reason, the syntax is:

```
[assembly:InternalsVisibleTo("TestAssembly1"),
    InternalsVisibleTo("TestAssembly2")]
```

Share Improve this answer Follow

edited Oct 3, 2008 at 16:40

answered Oct 3, 2008 at 16:31

cfeduke

23.2k ● 10 ● 63 ● 65

5

Decorate your method with this attribute:

```
[System.ComponentModel.EditorBrowsable(System.Componen
```

This will hide it from Intellisense.

EDIT:

But apparently this has a rather significant caveat: "In Visual C#, `EditorBrowsableAttribute` does not suppress members from a class in the same assembly." Via MSDN.

Share Improve this answer

Follow

edited Apr 29, 2013 at 6:19

Patrick D'Souza

3,553 ● 2 ● 24 ● 39

answered Oct 3, 2008 at 16:33

core

33k ● 45 ● 140 ● 195

> This sounded like the perfect solution, but it still shows up.
> – Mark T Oct 3, 2008 at 16:36

4

Suppose you want to test this object by manipulating its fields.

```
public class ComplexCalculation
{
    protected int favoriteNumber;
    public int FavoriteNumber
    {
        get { return favoriteNumber; }
    }
}
```

Place this object in your test assembly/namespace:

```
public class ComplexCalculationTest : ComplexCalculati
{
    public void SetFavoriteNumber(int newFavoriteNumbe
    {
        this.favoriteNumber = newFavoriteNumber;
    }
}
```

And write your test:

```
    public void Test()
    {
        ComplexCalculationTest myTestObject = new Comp
        myTestObject.SetFavoriteNumber(3);
        ComplexCalculation myObject = myTestObject;

        if (myObject.FavoriteNumber == 3)
            Console.WriteLine("Win!");

    }
```

PS: I know you said *internal*, but I don't think you meant internal.

Share Improve this answer

Follow

answered Oct 3, 2008 at 17:54

Amy B

110k ● 21 ● 139 ● 190

3

It sounds like your real problem is in your reference documents. You shouldn't test cases that are impossible to encounter under proper use of your class. If users shouldn't be allowed to change the state of those variables, then neither should your tests.

Share Improve this answer

Follow

answered Oct 3, 2008 at 16:36

Bill the Lizard

405k ● 211 ● 572 ● 889

> One cannot change scientific source documents. They are what they are. They are written by the international experts in the field who are not concerned about programming (nor should they have to be). – Mark T Oct 3, 2008 at 16:42

1

You can also use reflection. Google search turned up Unit testing private methods using reflection.

Share Improve this answer

Follow

answered Oct 3, 2008 at 17:53

Jon Adams

25.1k ● 18 ● 84 ● 121

0

Can your test code include a subclass of the calculations class? If so, you can mark the function `protected` and only inheritors will be able to use it. I'm pretty sure this also takes it out of intellisense, but I could be wrong about that.

Share Improve this answer

Follow

answered Oct 3, 2008 at 16:32

Ryan

9,928 ● 7 ● 44 ● 57

0

What I've done in the past is I put XML Comments by the method and used the section to write in big bold letters. DON'T USE THIS METHOD or whatever. That way, if someone tried to use it, Intellisense would give them a nice warning.

Share Improve this answer

Follow

answered Oct 3, 2008 at 16:42

BFree

104k ● 21 ● 162 ● 204