

Identifiable Object.cs

```
using System;  
using System.Collections.Generic;  
using System.Linq;  
using System.Runtime.CompilerServices;  
using System.Text;  
using System.Threading.Tasks;
```

namespace SwinAdventure

```
{  
    public class IdentifiableObject  
    {  
        private List<string> _identifiers;  
  
        public IdentifiableObject(string[] idents)  
        {  
            _identifiers = new List<string>();  
            foreach (string id in idents)  
            {  
                AddIdentifier(id);  
            }  
        }  
  
        public bool AreYou(string id)  
        {  
            return _identifiers.Contains(id.ToLower());  
        }  
    }  
}
```

```

public string FirstId
{
    get
    {
        if (_identifiers.Count > 0)
        {
            return _identifiers.First();
        }
        else
        {
            return string.Empty;
        }
    }
}

```

```

public void AddIdentifier(string id)
{
    _identifiers.Add(id.ToLower());
}
}

```

Unit testing

using NUnit.Framework;

using SwinAdventure;

namespace UnitTesting1

```

{
    [TestFixture]
    public class IdentifiableObjectTest

```

```
{  
    [Test]  
    public void TestAreYou()  
    {  
  
        string[] ids = { "fred", "bob" };  
        IdentifiableObject obj = new IdentifiableObject(ids);  
  
        bool resultFred = obj.AreYou("fred");  
        bool resultBob = obj.AreYou("bob");  
        bool resultWilma = obj.AreYou("wilma");  
  
        Assert.That(resultFred, Is.True);  
        Assert.That(resultBob, Is.True);  
        Assert.That(resultWilma, Is.False);  
    }  
}
```

```
[Test]  
public void TestNotAreYou()  
{  
  
    string[] ids = { "fred", "bob" };  
    IdentifiableObject obj = new IdentifiableObject(ids);  
  
    bool resultWilma = obj.AreYou("wilma");  
    bool resultBoby = obj.AreYou("boby");  
}
```

```
Assert.That(resultWilma, Is.False);  
Assert.That(resultBoby, Is.False);  
}
```

```
[Test]  
public void TestCaseSensitive()  
{  
  
    string[] ids = { "fred", "bob" };  
    IdentifiableObject obj = new IdentifiableObject(ids);  
  
    bool resultFredUpper = obj.AreYou("FRED");  
    bool resultBobLower = obj.AreYou("bOB");  
  
    Assert.That(resultFredUpper, Is.True); // Check case insensitivity for "fred"  
    Assert.That(resultBobLower, Is.True); // Check case insensitivity for "bob"  
}
```

```
[Test]  
public void TestFirstId()  
{  
  
    string[] ids = { "fred", "bob" };  
    IdentifiableObject obj = new IdentifiableObject(ids);  
  
    string firstId = obj.FirstId;
```

```

        Assert.That(firstId, Is.EqualTo("fred"));
    }

[Test]
public void TestFirstIdWithNolds()
{

    IdentifiableObject obj = new IdentifiableObject(new string[] { });

    string firstId = obj.FirstId;

    Assert.That(firstId, Is.EqualTo(string.Empty));
}

[Test]
public void TestAddId()
{
    // Arrange
    string[] ids = { "fred", "bob" };
    IdentifiableObject obj = new IdentifiableObject(ids);
    obj.AddIdentifier("wilma");

    bool resultWilma = obj.AreYou("wilma");

    // Assert
    Assert.That(resultWilma, Is.True);
}
}

```

}

Unit Test output

The screenshot shows the Test Explorer window in Visual Studio. The title bar reads "Test Explorer". The toolbar includes icons for running tests, a search bar labeled "Search (Ctrl+I)", and a summary bar showing "6" tests, "6" passed, and "0" failed. Below the toolbar, a status bar indicates "Test run finished: 6 Tests (6 Passed, 0 Failed, 0 Skipped) run in 626 ms" and "0 Warnings 0 Errors". The main table lists the test results:

Test	Duration	Traits	Error Message
UnitTesting1 (6)	388 ms		

On the right side, the "Group Summary" for "UnitTesting1" is displayed, showing "Tests in group: 6" and "Total Duration: 388 ms". Below this, the "Outcomes" section shows "6 Passed" with a green checkmark icon.