Swinburne University of Technology

COS20007 Object Oriented Programming

Case Study Iteration 1 - Identifiable Object

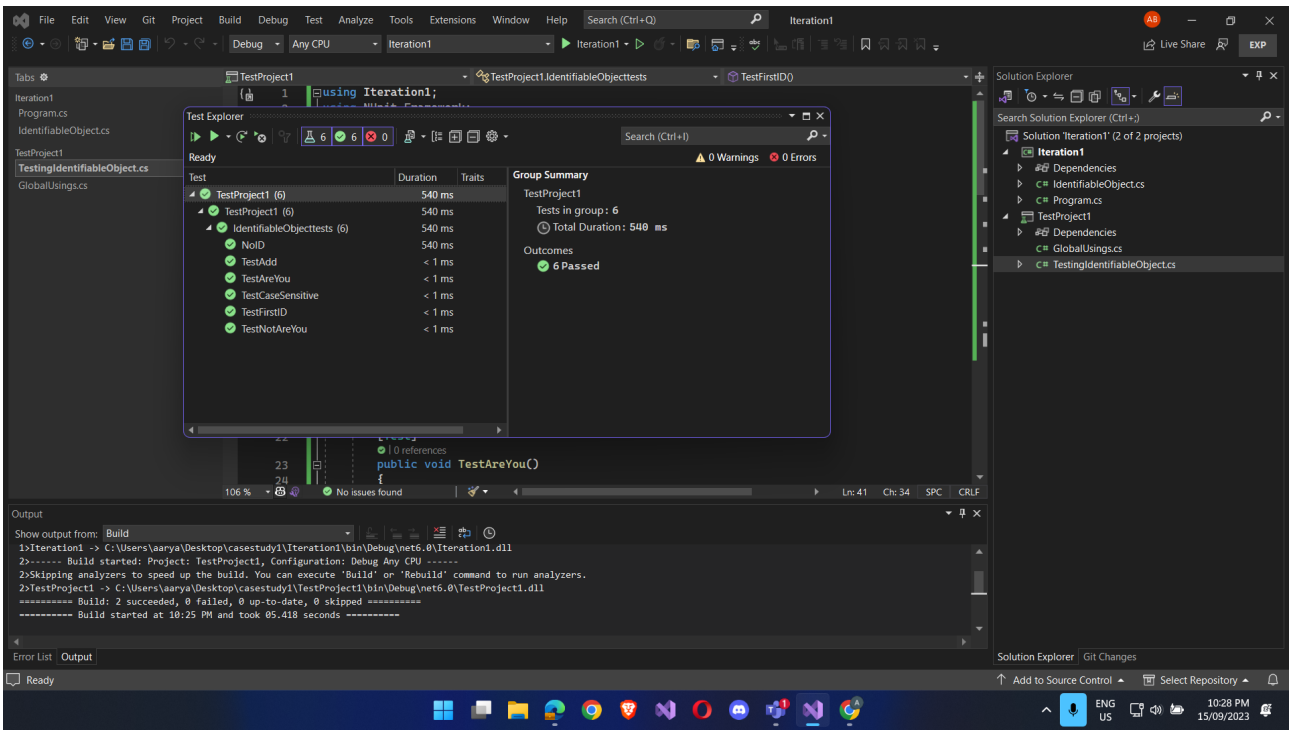
PDF generated at 12:17 on Saturday 16th September, 2023

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

namespace Iteration1
{
    // This class repreasents an object that can be identified by one or more identifiers
    public class IdentifiableObject
    {
        // To store list of identifiers

        private List<string> _identifiers;

        //_identifiers = new List<string>();

        // Constructor that initialises the IdentifiableObject with a array of identifiers

        public IdentifiableObject(string[] idents)
        {
            _identifiers = new List<string>(idents);
            foreach (string id in idents)
            {
                _identifiers.Add(id);
            }

        }


        // method to check if the object can be identified by a given identifier

        public bool AreYou(string id)
        {
            // lower case version of the input identifier
            return _identifiers.Contains(id.ToLower());
        }


        // property to return the first identifier in the list
        public string FirstId
        {
            get
            {
                if (_identifiers.Count > 0)
                {
                    return _identifiers[0];
                }
                else
                {
                    return String.Empty;
```

```
52                  }
53
54              }
55          }
56
57
58          // method to add an identifier to the list of identifiers
59          public void AddIdentifier(string id)
60          {
61              _identifiers.Add(id.ToLower());
62          }
63      }
64  }
```

```
using Iteration1;
using NUnit.Framework;
using System;

namespace TestProject1
{
    // Indicates this class contains tests
    [TestFixture]

    public class IdentifiableObjecttests
    {
        // declare a private field to hold the IdentifiableObject

        private IdentifiableObject id;

        // declare a private field to hold the IdentifiableObject with no identifiers or a just empty one

        private IdentifiableObject emptyId;



        [SetUp]
        public void SetUp()
        {
            // initialize the 'id' object with identifiers

            id = new IdentifiableObject(new string[] { "fred", "bob" });

            // initialize the 'emptyId' object with no identifiers

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


        [Test]
        public void TestAreYou()
        {
            // check if the object can be identified by a given identifier and then return true

            Assert.IsTrue(id.AreYou("fred"));
        }

        [Test] // test method
        public void TestNotAreYou()
        {
            // check if the object can be identified by a given identifier and then return false

            Assert.IsFalse(id.AreYou("wilma"));
        }

```

```
51          [Test]
52          public void TestCaseSensitive()
53          {
54              // check if the object can be identified (case sensitive) by a given
    ↪  identifier and then return true
55
56              Assert.IsTrue(id.AreYou("Fred"));
57          }
58
59          [Test]
60          public void TestFirstID()
61          {
62              //check if the 'id' object's  firstID property and then return fred
63
64              Assert.That(id.FirstId, Is.EqualTo("fred"));
65          }
66
67          [Test]
68          public void NoID()
69          {
70              // check if the 'emptyId' object's  firstID property and then return
    ↪  empty string
71
72              Assert.That(emptyId.FirstId, Is.EqualTo(""));
73          }
74          [Test]
75          public void TestAdd()
76          {
77              // Create a new IdentifiableObject object with no identifiers
78
79              id = new IdentifiableObject(new string[] { "fred", "bob" });
80              IdentifiableObject identifiableObject = new IdentifiableObject(new
    ↪  string[] { });
81
82              // add the "wilema" identifier to the 'id' object
83
84              id.AddIdentifier("wilma");
85
86              // check if the "id" object can be identified as the following and then
    ↪  then return true.
87              Assert.IsTrue(id.AreYou("bob"));
88              Assert.IsTrue(id.AreYou("wilma"));
89              Assert.IsTrue(id.AreYou("fred"));
90          }
91
92      }
93  }
```


Test Explorer
Ready
0 Warnings
0 Errors
Test
Duration
Traits
TestProject1 (6) 540 ms
TestProject1 (6) 540 ms
IdentifiableObjecttests (6) 540 ms
NoID 540 ms
TestAdd < 1 ms
TestAreYou < 1 ms
TestCaseSensitive < 1 ms
TestFirstID < 1 ms
TestNotAreYou < 1 ms
Group Summary
TestProject1
Tests in group: 6
Total Duration: 540 ms
Outcomes
6 Passed
using Iteration1;
0 references
public void TestAreYou()
{
Output
Show output from: Build
1>Iteration1 -> C:\Users\aarya\Desktop\casestudy1\Iteration1\bin\Debug\net6.0\Iteration1.dll
2>------ Build started: Project: TestProject1, Configuration: Debug Any CPU ------
2>Skipping analyzers to speed up the build. You can execute 'Build' or 'Rebuild' command to run analyzers.
2>TestProject1 -> C:\Users\aarya\Desktop\casestudy1\TestProject1\bin\Debug\net6.0\TestProject1.dll
========== Build: 2 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
---------- Build started at 10:25 PM and took 05.418 seconds ----------
Solution Explorer
Solution 'Iteration1' (2 of 2 projects)
Iteration1
Dependencies
IdentifiableObject.cs
Program.cs
TestProject1
Dependencies
GlobalUsings.cs
TestingIdentifiableObject.cs
Ready
Add to Source Control
Select Repository
10:28 PM
15/09/2023