Swinburne University of Technology

COS20007 Object Oriented Programming

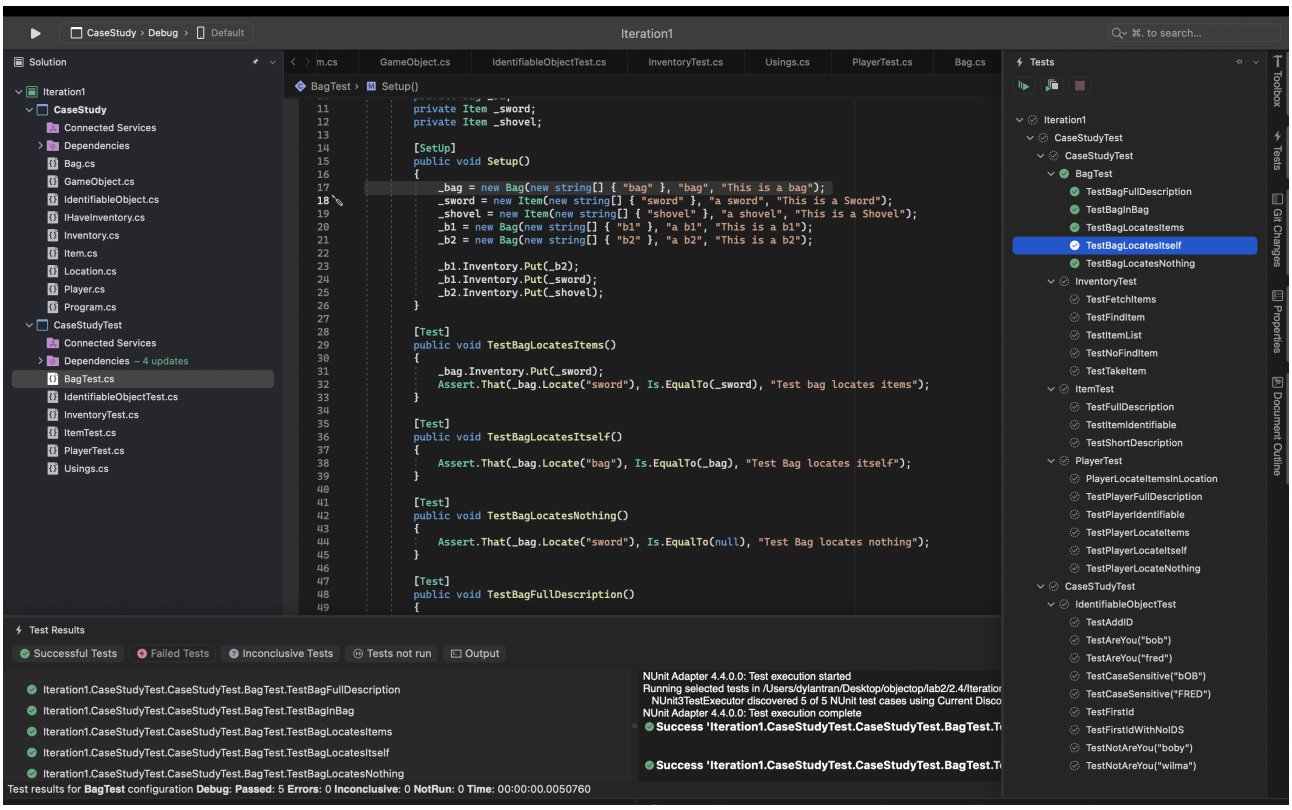
Case Study - Iteration 3 - Bags

PDF generated at 19:43 on Wednesday 4th October, 2023

```
using System;
namespace CaseStudy
{
        public class Bag : Item, IHaveInventory
        {
            private Inventory _inventory;

            public Bag(string[] ids, string name, string desc) : base(ids, name,
 ↪  desc)
            {
                _inventory = new Inventory();
            }

            public GameObject Locate(string id)
            {
                if (AreYou(id))
                {
                    return this;
                }
                else if (_inventory.HasItem(id))
                {
                    return _inventory.Fetch(id);
                }
                else
                {
                    return null;
                }
            }

            public override string FullDescription
            {
                get
                {
                    return $"In the {Name} you can see:\n" + _inventory.ItemList;
                }
            }

            public Inventory Inventory
            {
                get
                {
                    return _inventory;
                }
            }

        }
}

```

```
using System;
using CaseStudy;

namespace CaseStudyTest
{
    public class BagTest
    {
        private Bag _bag;
        private Bag _b1;
        private Bag _b2;
        private Item _sword;
        private Item _shovel;

        [SetUp]
        public void Setup()
        {
            _bag = new Bag(new string[] { "bag" }, "bag", "This is a bag");
            _sword = new Item(new string[] { "sword" }, "a sword", "This is a
↪   Sword");
            _shovel = new Item(new string[] { "shovel" }, "a shovel", "This is a
↪   Shovel");
            _b1 = new Bag(new string[] { "b1" }, "a b1", "This is a b1");
            _b2 = new Bag(new string[] { "b2" }, "a b2", "This is a b2");

            _b1.Inventory.Put(_b2);
            _b1.Inventory.Put(_sword);
            _b2.Inventory.Put(_shovel);
        }

        [Test]
        public void TestBagLocatesItems()
        {
            _bag.Inventory.Put(_sword);
            Assert.That(_bag.Locate("sword"), Is.EqualTo(_sword), "Test bag locates
↪   items");
        }

        [Test]
        public void TestBagLocatesItself()
        {
            Assert.That(_bag.Locate("bag"), Is.EqualTo(_bag), "Test Bag locates
↪   itself");
        }

        [Test]
        public void TestBagLocatesNothing()
        {
            Assert.That(_bag.Locate("sword"), Is.EqualTo(null), "Test Bag locates
↪   nothing");
        }

        [Test]
        public void TestBagFullDescription()
```

```
49          {
50              Assert.That(_bag.FullDescription, Is.EqualTo("In the bag you can see:\n"
    ↪  + _bag.Inventory.ItemList), "Test Bag Full Description");
51          }
52
53          [Test]
54          public void TestBagInBag()
55          {
56              Assert.That(_b1.Locate("b2"), Is.EqualTo(_b2), "Test b1 can locate b2");
57
58              Assert.That(_b1.Locate("sword"), Is.EqualTo(_sword), "Test b1 can locate
    ↪  other items in b1");
59
60              Assert.That(_b1.Locate("shovel"), Is.EqualTo(null), "Test b1 can not
    ↪  locate items in b2");
61          }
62      }
63  }
64
65
```


Iteration1
Solution
Iteration1
CaseStudy
Connected Services
Dependencies
Bag.cs
GameObject.cs
IdentifiableObject.cs
IHaveInventory.cs
Inventory.cs
Item.cs
Location.cs
Player.cs
Program.cs
CaseStudyTest
Connected Services
Dependencies – 4 updates
BagTest.cs
IdentifiableObjectTest.cs
InventoryTest.cs
ItemTest.cs
PlayerTest.cs
Usings.cs
BagTest › Setup()
private Item _sword;
private Item _shovel;
[SetUp]
public void Setup()
{
_bag = new Bag(new string[] { "bag" }, "bag", "This is a bag");
_sword = new Item(new string[] { "sword" }, "a sword", "This is a Sword");
_shovel = new Item(new string[] { "shovel" }, "a shovel", "This is a Shovel");
_b1 = new Bag(new string[] { "b1" }, "a b1", "This is a b1");
_b2 = new Bag(new string[] { "b2" }, "a b2", "This is a b2");
_b1.Inventory.Put(_b2);
_b1.Inventory.Put(_sword);
_b2.Inventory.Put(_shovel);
}
[Test]
public void TestBagLocatesItems()
{
_bag.Inventory.Put(_sword);
Assert.That(_bag.Locate("sword"), Is.EqualTo(_sword), "Test bag locates items");
}
[Test]
public void TestBagLocatesItself()
{
Assert.That(_bag.Locate("bag"), Is.EqualTo(_bag), "Test Bag locates itself");
}
[Test]
public void TestBagLocatesNothing()
{
Assert.That(_bag.Locate("sword"), Is.EqualTo(null), "Test Bag locates nothing");
}
[Test]
public void TestBagFullDescription()
{
Test Results
Successful Tests
Failed Tests
Inconclusive Tests
Tests not run
Output
Iteration1.CaseStudyTest.CaseStudyTest.BagTest.TestBagFullDescription
Iteration1.CaseStudyTest.CaseStudyTest.BagTest.TestBagInBag
Iteration1.CaseStudyTest.CaseStudyTest.BagTest.TestBagLocatesItems
Iteration1.CaseStudyTest.CaseStudyTest.BagTest.TestBagLocatesItself
Iteration1.CaseStudyTest.CaseStudyTest.BagTest.TestBagLocatesNothing
Test results for BagTest configuration Debug: Passed: 5 Errors: 0 Inconclusive: 0 NotRun: 0 Time: 00:00:00.0050760
NUnit Adapter 4.4.0.0: Test execution started
NUnit Adapter 4.4.0.0: Test execution complete
Tests
Iteration1
CaseStudyTest
BagTest
TestBagFullDescription
TestBagInBag
TestBagLocatesItems
TestBagLocatesItself
TestBagLocatesNothing
InventoryTest
TestFetchItems
TestFindItem
TestItemList
TestNoFindItem
TestTakeItem
ItemTest
TestFullDescription
TestItemIdentifiable
TestShortDescription
PlayerTest
PlayerLocateItemsInLocation
TestPlayerFullDescription
TestPlayerIdentifiable
TestPlayerLocateItems
TestPlayerLocateItself
TestPlayerLocateNothing
CaseSTudyTest
IdentifiableObjectTest
TestAddID
TestAreYou("bob")
TestAreYou("fred")
TestCaseSensitive("bOB")
TestCaseSensitive("FRED")
TestFirstId
TestFirstIdWithNoIDS
TestNotAreYou("boby")
TestNotAreYou("wilma")