Swinburne University of Technology

COS20007 Object Oriented Programming

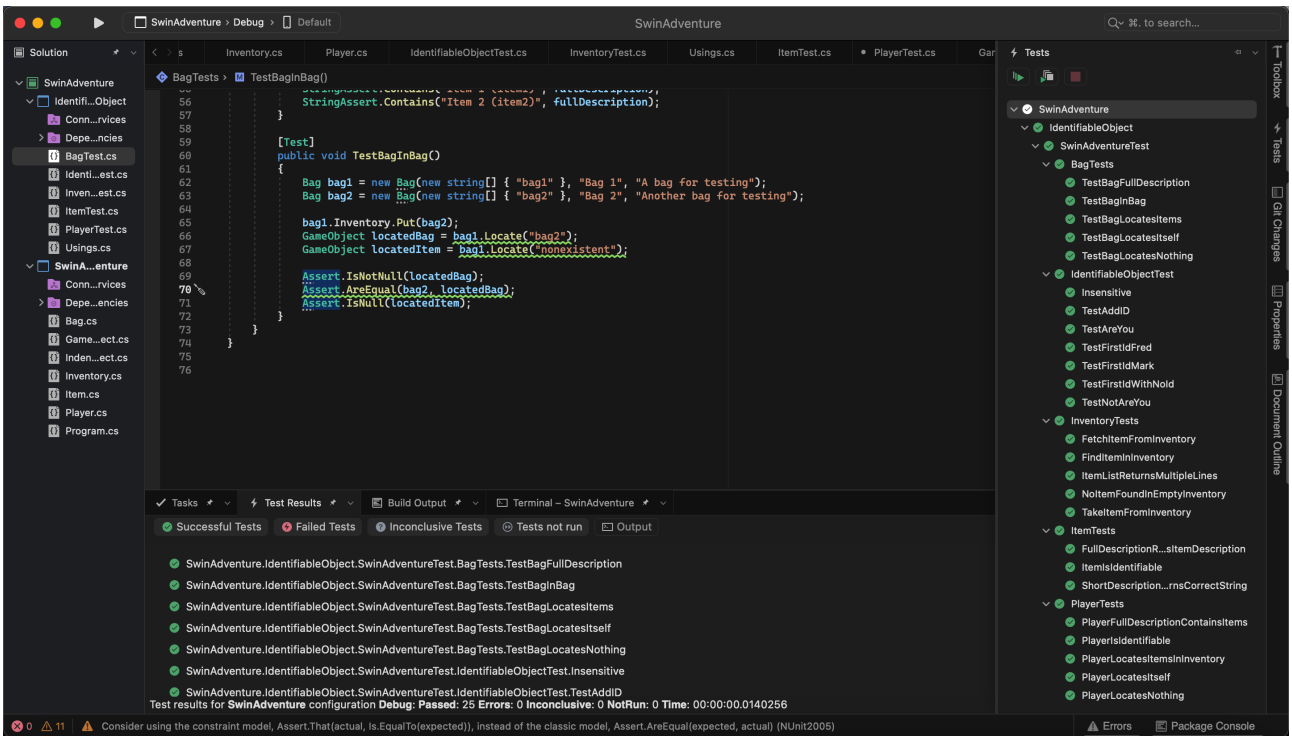
Case Study - Iteration 3 - Bags

PDF generated at 05:34 on Friday 6th October, 2023

```
using System;
using System.Collections.Generic;

namespace SwinAdventure
{
    public class Bag : Item
    {
        private Inventory _inventory;

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

        public Inventory Inventory
        {
            get { return _inventory; }
        }

        public override string FullDescription
        {
            get
            {
                string bagDesc = "In the " + Name + " you can see:" +
    ↪ Environment.NewLine;
                bagDesc += _inventory.ItemList;
                return bagDesc;
            }
        }

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

```

```
using System;
using NUnit.Framework;
using SwinAdventure;

namespace SwinAdventureTest
{
    public class BagTests
    {
        [Test]
        public void TestBagLocatesItems()
        {
            Bag bag = new Bag(new string[] { "bag1" }, "Bag 1", "A bag for testing");
            Item item = new Item(new string[] { "item1" }, "Item 1", "An item for
↪ testing");

            bag.Inventory.Put(item);
            Item locatedItem = (Item)bag.Locate("item1");

            Assert.IsNotNull(locatedItem);
            Assert.IsTrue(bag.Inventory.HasItem("item1"));
        }

        [Test]
        public void TestBagLocatesItself()
        {
            Bag bag = new Bag(new string[] { "bag1" }, "Bag 1", "A bag for testing");

            GameObject locatedObject = bag.Locate("bag1");

            Assert.IsNotNull(locatedObject);
            Assert.AreEqual(bag, locatedObject);
        }

        [Test]
        public void TestBagLocatesNothing()
        {
            Bag bag = new Bag(new string[] { "bag1" }, "Bag 1", "A bag for testing");

            GameObject locatedObject = bag.Locate("nonexistent");

            Assert.IsNull(locatedObject);
        }

        [Test]
        public void TestBagFullDescription()
        {
            Bag bag = new Bag(new string[] { "bag1" }, "Bag 1", "A bag for testing");
            Item item1 = new Item(new string[] { "item1" }, "Item 1", "An item for
↪ testing");
            Item item2 = new Item(new string[] { "item2" }, "Item 2", "Another item
↪ for testing");

            bag.Inventory.Put(item1);
```

```
51              bag.Inventory.Put(item2);
52              string fullDescription = bag.FullDescription;
53
54              StringAssert.Contains("In the Bag 1 you can see:", fullDescription);
55              StringAssert.Contains("Item 1 (item1)", fullDescription);
56              StringAssert.Contains("Item 2 (item2)", fullDescription);
57          }
58
59          [Test]
60          public void TestBagInBag()
61          {
62              Bag bag1 = new Bag(new string[] { "bag1" }, "Bag 1", "A bag for
    ↪  testing");
63              Bag bag2 = new Bag(new string[] { "bag2" }, "Bag 2", "Another bag for
    ↪  testing");
64
65              bag1.Inventory.Put(bag2);
66              GameObject locatedBag = bag1.Locate("bag2");
67              GameObject locatedItem = bag1.Locate("nonexistent");
68
69              Assert.IsNotNull(locatedBag);
70              Assert.AreEqual(bag2, locatedBag);
71              Assert.IsNull(locatedItem);
72          }
73      }
74  }
75
```


SwinAdventure › Debug › Default
SwinAdventure
Solution
SwinAdventure
Inventory.cs
Player.cs
IdentifiableObjectTest.cs
InventoryTest.cs
Usings.cs
ItemTest.cs
PlayerTest.cs
BagTests › TestBagInBag()
StringAssert.Contains("Item 2 (item2)", fullDescription);
}
[Test]
public void TestBagInBag()
{
Bag bag1 = new Bag(new string[] { "bag1" }, "Bag 1", "A bag for testing");
Bag bag2 = new Bag(new string[] { "bag2" }, "Bag 2", "Another bag for testing");
bag1.Inventory.Put(bag2);
GameObject locatedBag = bag1.Locate("bag2");
GameObject locatedItem = bag1.Locate("nonexistent");
Assert.IsNotNull(locatedBag);
Assert.AreEqual(bag2, locatedBag);
Assert.IsNull(locatedItem);
}
}
}
Tasks
Test Results
Build Output
Terminal – SwinAdventure
Successful Tests
Failed Tests
Inconclusive Tests
Tests not run
Output
SwinAdventure.IdentifiableObject.SwinAdventureTest.BagTests.TestBagFullDescription
SwinAdventure.IdentifiableObject.SwinAdventureTest.BagTests.TestBagInBag
SwinAdventure.IdentifiableObject.SwinAdventureTest.BagTests.TestBagLocatesItems
SwinAdventure.IdentifiableObject.SwinAdventureTest.BagTests.TestBagLocatesItself
SwinAdventure.IdentifiableObject.SwinAdventureTest.BagTests.TestBagLocatesNothing
SwinAdventure.IdentifiableObject.SwinAdventureTest.IdentifiableObjectTest.Insensitive
SwinAdventure.IdentifiableObject.SwinAdventureTest.IdentifiableObjectTest.TestAddID
Test results for SwinAdventure configuration Debug: Passed: 25 Errors: 0 Inconclusive: 0 NotRun: 0 Time: 00:00:00.0140256
Tests
SwinAdventure
IdentifiableObject
SwinAdventureTest
BagTests
TestBagFullDescription
TestBagInBag
TestBagLocatesItems
TestBagLocatesItself
TestBagLocatesNothing
IdentifiableObjectTest
Insensitive
TestAddID
TestAreYou
TestFirstIdFred
TestFirstIdMark
TestFirstIdWithNoId
TestNotAreYou
InventoryTests
FetchItemFromInventory
FindItemInInventory
ItemListReturnsMultipleLines
NoItemFoundInEmptyInventory
TakeItemFromInventory
ItemTests
FullDescriptionR...sItemDescription
ItemIsIdentifiable
ShortDescription...rnsCorrectString
PlayerTests
PlayerFullDescriptionContainsItems
PlayerIsIdentifiable
PlayerLocatesItemsInInventory
PlayerLocatesItself
PlayerLocatesNothing
Consider using the constraint model, Assert.That(actual, Is.EqualTo(expected)), instead of the classic model, Assert.AreEqual(expected, actual) (NUnit2005)
Errors
Package Console