Swinburne University of Technology

COS20007 Object Oriented Programming

Case Study - Iteration 3 - Bags

PDF generated at 02:05 on Friday 10th November, 2023

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

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

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

        public Inventory Inventory
        {
            get { return _inventory; }
        }

        public GameObject Locate(string id)
        {
            List<GameObject> list = new List<GameObject>();

            if(id == FirstId)
            {
                list.Add(this);
            }
            else if (_inventory.HasItem(id))
            {
                list.Add(_inventory.Fetch(id));
            }
            else
            {
                list.Add(null);
            }

            var result = list.ElementAt(0);
            list.Clear();
            return result;
        }
    }
}
```

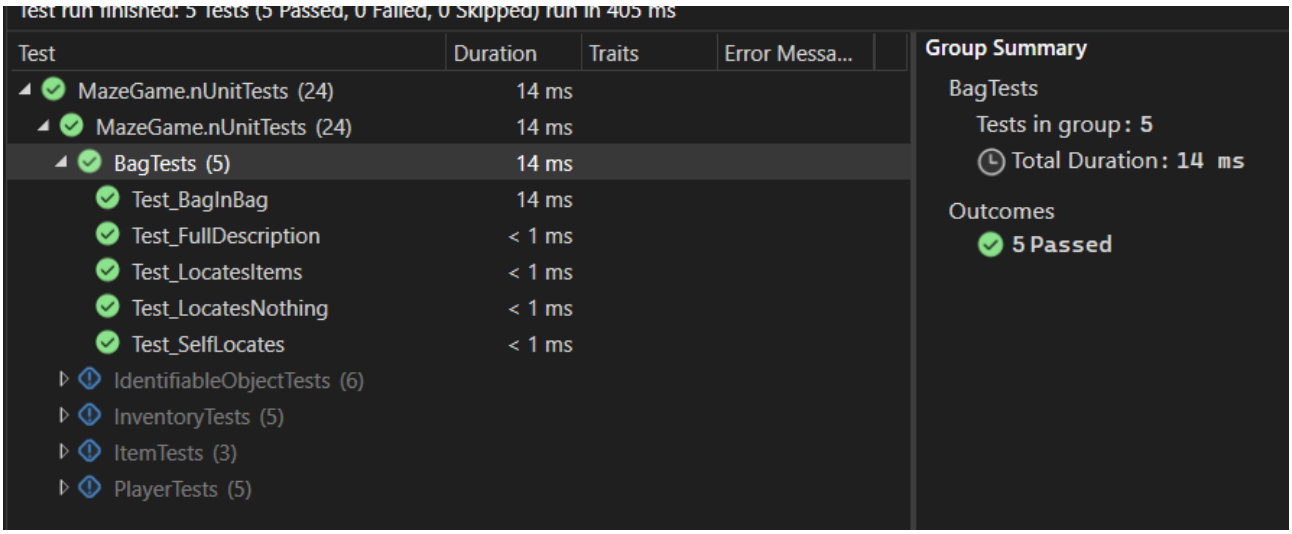
```
namespace MazeGame.nUnitTests
{
    public class BagTests
    {
        private Bag _bag1 { get; set; } = null!;
        private Item sword { get; set; } = null!;
        private Item shovel { get; set; } = null!;
        private Item pickaxe { get; set; } = null!;
        [SetUp]
        public void SetUp()
        {
            _bag1 = new Bag(new string[] { "b1" }, "brown bag", "a bag made and stiched with leather.");
            sword = new Item(new string[] { "sword" }, "a bronze sword", "A short sword cast from bronze");
            shovel = new Item(new string[] { "shovel" }, "a shovel", "A durable shovel borrowed from the village");
            pickaxe = new Item(new string[] { "pickaxe" }, "an obsidian pickaxe", "A pickaxe made of obsidian");
            _bag1.Inventory.Put(sword);
            _bag1.Inventory.Put(shovel);
            _bag1.Inventory.Put(pickaxe);
        }

        [Test]
        public void Test_LocatesItems()
        {
            string sampleID = "sword";
            var sut = _bag1.Locate(sampleID);
            Assert.That(sut, Is.EqualTo(sword));
            Console.WriteLine(sut.ShortDescription);
        }

        [Test]
        public void Test_SelfLocates()
        {
            string sampleID = "b1";
            var sut = _bag1.Locate(sampleID);
            Assert.That(sut, Is.EqualTo(_bag1));
            Console.WriteLine(sut.ShortDescription);
        }

        [Test]
        public void Test_LocatesNothing()
        {
            string samepleID = "b123";
            var sut = _bag1.Locate(samepleID);
            Assert.IsNull(sut);
            if (sut == null)
            {
                Console.WriteLine("Bag returned null value");
            }
        }
```

```
50
51          [Test]
52          public void Test_FullDescription()
53          {
54              var sut = _bag1.FullDescription;
55              Assert.IsNotNull(sut);
56              Console.WriteLine(sut);
57          }
58
59          [Test]
60          public void Test_BagInBag()
61          {
62              Bag _bag2 = new Bag(new string[] { "b2" }, "backpack", "A bag with high
    ↪  durability and storage capacity");
63              Item bat = new Item(new string[] { "bat" }, "a baseball bat", "A baseball
    ↪  bat made of wood");
64              Item flashlight = new Item(new string[] { "flashlight" }, "a flashlight",
    ↪  "A UV flashlight that shines brightly at night");
65              _bag2.Inventory.Put(bat);
66              _bag2.Inventory.Put(flashlight);
67              _bag1.Inventory.Put(_bag2);
68
69              var sut1 = _bag1.Locate("b2");
70              Assert.IsNotNull(sut1);
71              Console.WriteLine("Located bag2 in bag1: {0}", sut1.ShortDescription);
72
73              var sut2 = _bag1.Locate("shovel");
74              Assert.That(sut2.Name, Is.EqualTo("a shovel"));
75              Console.WriteLine("Locate existing item in bag1: {0}",
    ↪  sut2.ShortDescription );
76
77              var sut3 = _bag1.Locate("flashlight");
78              Assert.IsNull(sut3);
79              if(sut3 == null)
80              {
81                  Console.WriteLine("bag2's items can not be found in bag1");
82              }
83          }
84      }
85  }
```


Test run finished: 5 Tests (5 Passed, 0 Failed, 0 Skipped) run in 405 ms
Test
Duration
Traits
Error Messa...
MazeGame.nUnitTests (24) 14 ms
MazeGame.nUnitTests (24) 14 ms
BagTests (5) 14 ms
Test_BagInBag 14 ms
Test_FullDescription < 1 ms
Test_LocatesItems < 1 ms
Test_LocatesNothing < 1 ms
Test_SelfLocates < 1 ms
IdentifiableObjectTests (6)
InventoryTests (5)
ItemTests (3)
PlayerTests (5)
Group Summary
BagTests
Tests in group: 5
Total Duration: 14 ms
Outcomes
5 Passed