Swinburne University of Technology

COS20007 Object Oriented Programming

Case Study - Iteration 2 - Players Items and Inventory

PDF generated at 17:04 on Sunday 24th September, 2023

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

namespace SwinAdventure
{
    public abstract class GameObject : IdentifiableObject
    {
        private string _description;
        private string _name;
        public GameObject(string[] ids, string name, string desc) : base(ids)
        {
            _description = desc;
            _name = name;
        }
        public string Name
        {
               get { return _name; }
        }
        public string ShortDescription
        {
            get
            {
                return $"{Name} ({FirstID})";
            }
        }
        public virtual string FullDescription
        {
            get { return _description; }
        }
    }
}
```

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

namespace SwinAdventure
{
    public class Player : GameObject
    {
        private Inventory _inventory;
        public Player(string name, string desc) : base(new string[] {"me",
    ↪  "inventory"}, name, desc)
        {
            _inventory = new Inventory();
        }
        public GameObject Locate(string id)
        {
            if (AreYou(id)) return this;
            return _inventory.Fetch(id);
        }
        public override string FullDescription
        {
        get
            {
                return $"You are {Name}, You are carrying: {_inventory.ItemList}";
            }
        }
        public Inventory Inventory { get { return _inventory; } }
    }
}
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using SwinAdventure;
using NUnit.Framework;

namespace SwinAdventureTests
{
    [TestFixture]
    public class PlayerTest
    {
        private Player _playerObject;
        private Item _shovel;
        private Item _sword;

        [SetUp]
        public void SetUp()
        {
            _playerObject = new Player("Vinh", "SwinAdventure");
            _shovel = new Item(new string[] { "shovel" }, "a shovel", "This is a shovel");
            _sword = new Item(new string[] { "sword" }, "a sword", "This is a Sword");
        }

        [Test]
        public void TestPlayerIsIdentifiable()
        {
            Assert.IsTrue(_playerObject.AreYou("me") && _playerObject.AreYou("inventory"));
        }

        [Test]
        public void TestPlayerLocatesItems()
        {
            _playerObject.Inventory.Put(_shovel);
            Assert.AreEqual(_shovel, _playerObject.Locate("shovel"));
        }

        [Test]
        public void TestPlayerLocatesItself()
        {
            Assert.AreEqual(_playerObject, _playerObject.Locate("me"));
            Assert.AreEqual(_playerObject, _playerObject.Locate("inventory"));
        }

        [Test]
        public void TestPlayerLocateNothing()
        {
            Assert.IsNull(_playerObject.Locate("hi"));
        }
```

```
51
52          [Test]
53          public void TestPlayerFullDescription()
54          {
55              _playerObject.Inventory.Put(_sword);
56              _playerObject.Inventory.Put(_shovel);
57              string expected = $"You are {_playerObject.Name}, You are carrying:
    ↪  {_playerObject.Inventory.ItemList}";
58              Assert.AreEqual(expected, _playerObject.FullDescription);
59          }
60      }
61  }
```

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

namespace SwinAdventure
{
    public class Item : GameObject
    {
        public Item(string[] idents, string name, string desc) : base(idents, name,
↪   desc)
        {

        }
    }
}
```

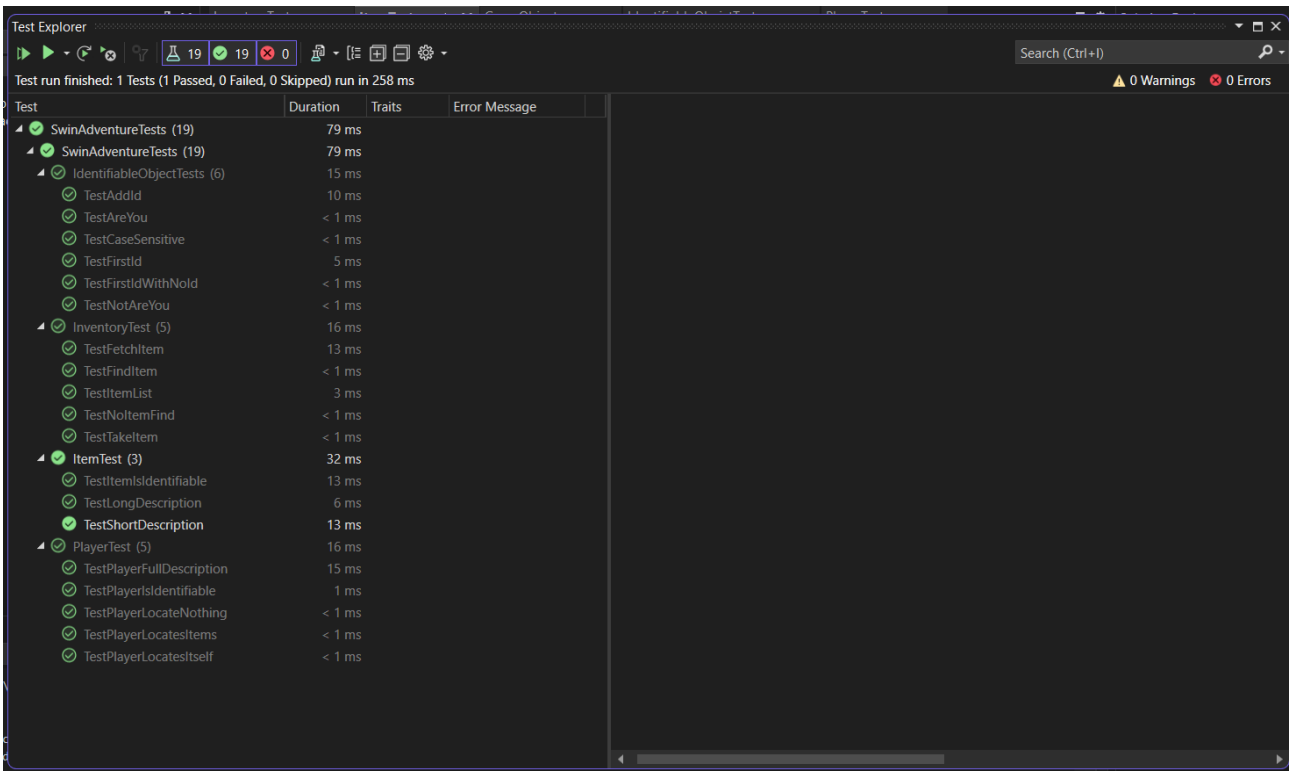
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using SwinAdventure;
using NUnit.Framework;

namespace SwinAdventureTests
{
    [TestFixture]
    public class ItemTest
    {
        private Item _itemObject;

        [SetUp]
        public void SetUp()
        {
            _itemObject = new Item(new string[] { "Bronze Sword", "The basic sword" }, "Bronze Sword", "Bronze Sword is the first sword that players will obtain after finishing the first quest");
        }

        [Test]
        public void TestItemIsIdentifiable()
        {
            Assert.IsTrue(_itemObject.AreYou("Bronze Sword"));
        }

        [Test]
        public void TestShortDescription()
        {
            Assert.AreEqual("Bronze Sword (bronze sword)", _itemObject.ShortDescription);
        }

        [Test]
        public void TestLongDescription()
        {
            Assert.AreEqual("Bronze Sword is the first sword that players will obtain after finishing the first quest", _itemObject.FullDescription);
        }
    }
}
```

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

namespace SwinAdventure
{
    public class Inventory
    {
        private List<Item> _items;
        public Inventory()
        {
            _items = new List<Item>();
        }
        public bool HasItem(string id)
        {
            foreach(Item itm in _items)
            {
                if(itm.AreYou(id)) return true;
            }
            return false;
        }
        public void Put(Item itm)
        {
            _items.Add(itm);
        }
        public Item Take(string id)
        {
            Item takeItem = this.Fetch(id);
                _items.Remove(takeItem);
            return takeItem;
        }
        public Item Fetch(string id)
        {
            foreach (Item itm in _items)
            {
                if (itm.AreYou(id)) return itm;
            }
            return null;
        }
        public string ItemList
        {
            get
            {
                string listItem = "";
                foreach(Item itm in _items)
                {
                    listItem = listItem + itm.ShortDescription + "\n";
                }
                return listItem;
            }
        }
```

```
54      }
55  }
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using SwinAdventure;
using NUnit.Framework;

namespace SwinAdventureTests
{
    [TestFixture]
    public class InventoryTest
    {
        private Inventory _inventoryObject;
        private Item _shovel;
        private Item _sword;

        [SetUp]
        public void SetUp()
        {
            _inventoryObject = new Inventory();
            _shovel = new Item(new string[] { "shovel" }, "a shovel", "This is a
↪   shovel");
            _sword = new Item(new string[] { "sword" }, "a sword", "This is a
↪   Sword");
        }

        [Test]
        public void TestFindItem()
        {
            _inventoryObject.Put(_shovel);
            Assert.IsTrue(_inventoryObject.HasItem("shovel"));
        }

        [Test]
        public void TestNoItemFind()
        {
            _inventoryObject.Put(_shovel);
            Assert.IsFalse(_inventoryObject.HasItem("sword"));
        }

        [Test]
        public void TestFetchItem()
        {
            _inventoryObject.Put(_shovel);
            Assert.AreEqual(_shovel, _inventoryObject.Fetch("shovel"));
            Assert.IsTrue(_inventoryObject.HasItem("shovel"));
        }

        [Test]
        public void TestTakeItem()
        {
            _inventoryObject.Put(_shovel);
```

```
52              _inventoryObject.Take("shovel");
53              Assert.IsFalse(_inventoryObject.HasItem("shovel"));
54          }
55
56          [Test]
57          public void TestItemList()
58          {
59              _inventoryObject.Put(_shovel);
60              _inventoryObject.Put(_sword);
61              Assert.IsTrue(_inventoryObject.HasItem("shovel"));
62              Assert.IsTrue(_inventoryObject.HasItem("sword"));
63
64              string expectedOutput = "a shovel (shovel)\n" + "a sword (sword)\n";
65              Assert.AreEqual(expectedOutput, _inventoryObject.ItemList);
66          }
67      }
68  }
```


Test Explorer
19
19
0
Search (Ctrl+I)
Test run finished: 1 Tests (1 Passed, 0 Failed, 0 Skipped) run in 258 ms
0 Warnings
0 Errors
Test
Duration
Traits
Error Message
SwinAdventureTests (19) 79 ms
SwinAdventureTests (19) 79 ms
IdentifiableObjectTests (6) 15 ms
TestAddId 10 ms
TestAreYou < 1 ms
TestCaseSensitive < 1 ms
TestFirstId 5 ms
TestFirstIdWithNoId < 1 ms
TestNotAreYou < 1 ms
InventoryTest (5) 16 ms
TestFetchItem 13 ms
TestFindItem < 1 ms
TestItemList 3 ms
TestNoItemFind < 1 ms
TestTakeItem < 1 ms
ItemTest (3) 32 ms
TestItemIsIdentifiable 13 ms
TestLongDescription 6 ms
TestShortDescription 13 ms
PlayerTest (5) 16 ms
TestPlayerFullDescription 15 ms
TestPlayerIsIdentifiable 1 ms
TestPlayerLocateNothing < 1 ms
TestPlayerLocatesItems < 1 ms
TestPlayerLocatesItself < 1 ms