Swinburne University of Technology

COS20007 Object Oriented Programming

Case Study - Iteration 2 - Players Items and Inventory

PDF generated at 12:08 on Tuesday 12th September, 2023

Produced by OOP Submission Formatter

```
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; } }
    }
}
```

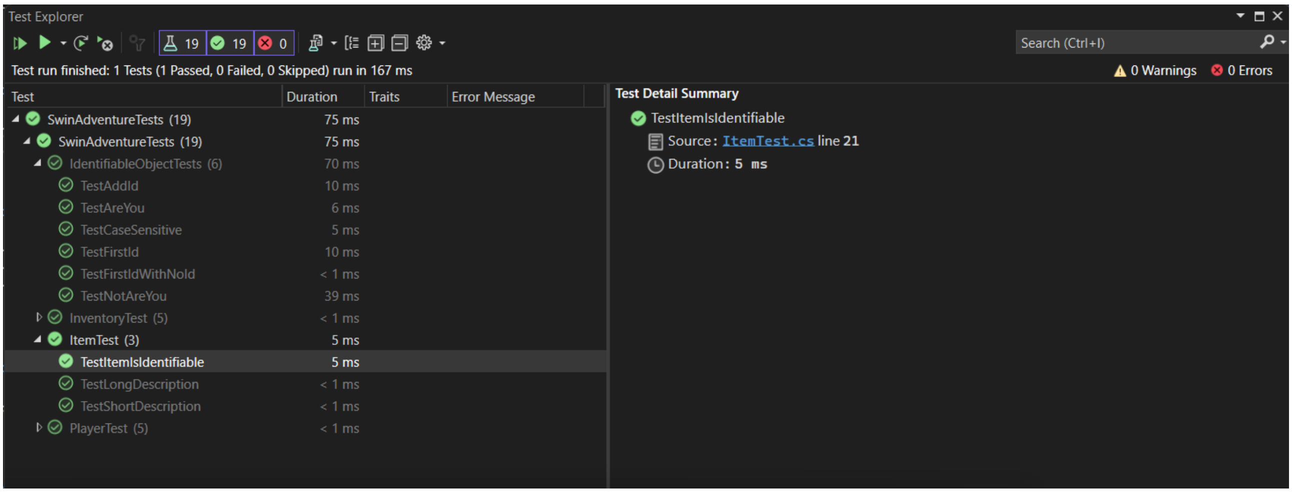
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Numerics;
using System.Text;
using System.Threading.Tasks;
using SwinAdventure;
namespace SwinAdventureTests
{
    public class PlayerTest
    {
        private Player _playerObject;
        private Item _shovel;
        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);
            var itemLocate = _playerObject.Locate("shovel");
            Assert.AreEqual(_shovel, itemLocate);
        }
        [Test]
        public void TestPlayerLocatesItself()
        {
            var me = _playerObject.Locate("me");
            var inv = _playerObject.Locate("inventory");

            Assert.IsTrue(me == _playerObject || inv == _playerObject);
        }
        [Test]
        public void TestPlayerLocateNothing()
        {
            var me = _playerObject.Locate("hi");
            Assert.IsNull(me);
        }
        [Test]
        public void TestPlayerFullDescription()
        {
            _playerObject.Inventory.Put(_sword);
            _playerObject.Inventory.Put(_shovel);
            string expected = $"You are {_playerObject.Name}, You are carrying: {_playerObject.Inventory.ItemList}";
            Assert.AreEqual(expected, _playerObject.FullDescription);
        }
    }
}
```

```
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;

namespace SwinAdventureTests
{
    public class ItemTest
    {
        private Item _itemObject;
        private string[] idents = new[] { "Bronze Sword", "The basic sword" };

        [SetUp]
        public void SetUp()
        {
            _itemObject = new Item(idents, "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(id: "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;
            }
        }
    }
}
```

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

namespace SwinAdventureTests
{
    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.FirstID));
        }
        [Test]
        public void TestNoItemFind()
        {
            _inventoryObject.Put(_shovel);
            Assert.IsFalse(_inventoryObject.HasItem(_sword.FirstID));
        }
        [Test]
        public void TestFetchItem()
        {
            _inventoryObject.Put(_shovel);
            Item fetchItem = _inventoryObject.Fetch(_shovel.FirstID);

            Assert.AreEqual(_shovel, fetchItem);
            Assert.IsTrue(_inventoryObject.HasItem(_shovel.FirstID));
        }
        [Test]
        public void TestTakeItem()
        {
            _inventoryObject.Put(_shovel);
            _inventoryObject.Take(_shovel.FirstID);

            Assert.IsFalse(_inventoryObject.HasItem(_shovel.FirstID));
        }
        [Test]
        public void TestItemList()
        {
            _inventoryObject.Put(_shovel);
            _inventoryObject.Put(_sword);
            Assert.IsTrue(_inventoryObject.HasItem(_shovel.FirstID));
            Assert.IsTrue(_inventoryObject.HasItem(_sword.FirstID));

            string expectOutput = "a shovel (shovel)\n" + "a sword (sword)\n";
            Assert.AreEqual(_inventoryObject.ItemList, expectOutput);
        }
    }
}
```



Test run finished: 1 Tests (1 Passed, 0 Failed, 0 Skipped) run in 167 ms

⚠ 0 Warnings ⊗ 0 Errors

Test	Duration	Traits	Error Message
✓ SwinAdventureTests (19)	75 ms		
✓ SwinAdventureTests (19)	75 ms		
✓ IdentifiableObjectTests (6)	70 ms		
✓ TestAddId	10 ms		
✓ TestAreYou	6 ms		
✓ TestCaseSensitive	5 ms		
✓ TestFirstId	10 ms		
✓ TestFirstIdWithNoId	< 1 ms		
✓ TestNotAreYou	39 ms		
✓ InventoryTest (5)	< 1 ms		
✓ ItemTest (3)	5 ms		
✓ TestItemIsIdentifiable	5 ms		
✓ TestLongDescription	< 1 ms		
✓ TestShortDescription	< 1 ms		
✓ PlayerTest (5)	< 1 ms		

Test Detail Summary

✓ TestItemIsIdentifiable

Source: ItemTest.cs line 21

Duration: 5 ms