Swinburne University of Technology

COS20007 Object Oriented Programming

6.1P - Case Study - Iteration 4 - Look Command

PDF generated at 22:21 on Wednesday 5th April, 2023

```
using System;

namespace SwinAdventure
{
    public interface IHaveInventory
    {
        public GameObject Locate(string id);
        public string Name
        {
            get;
        }
    }
}
```

```
using System;

namespace SwinAdventure
{
    public class Player : GameObject, IHaveInventory
    {
        public Player(string name, string desc) : base( new string[] { "me",
↪   "inventory"}, name, desc)
        {
            _inventory = new Inventory();
        }

        private Inventory _inventory;

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

        public override string FullDescription
        {
            get { return $"You are {Name}, ({base.FullDescription}), you are
↪   carrying:\n" + _inventory.ItemList; }
        }

        public Inventory Inventory
        {
            get { return _inventory; }
        }
    }
}
```

```
namespace SwinAdventure
{
    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);
            return null;
        }

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

        public Inventory Inventory
        {
            get { return _inventory; }
        }
    }
}
```

```
using System;

namespace SwinAdventure
{
    public abstract class Command : IdentifiableObject
    {
        public Command(string[] ids) : base(ids)
        {
        }

        public abstract string Execute(Player p, string[] text);
    }
}
```

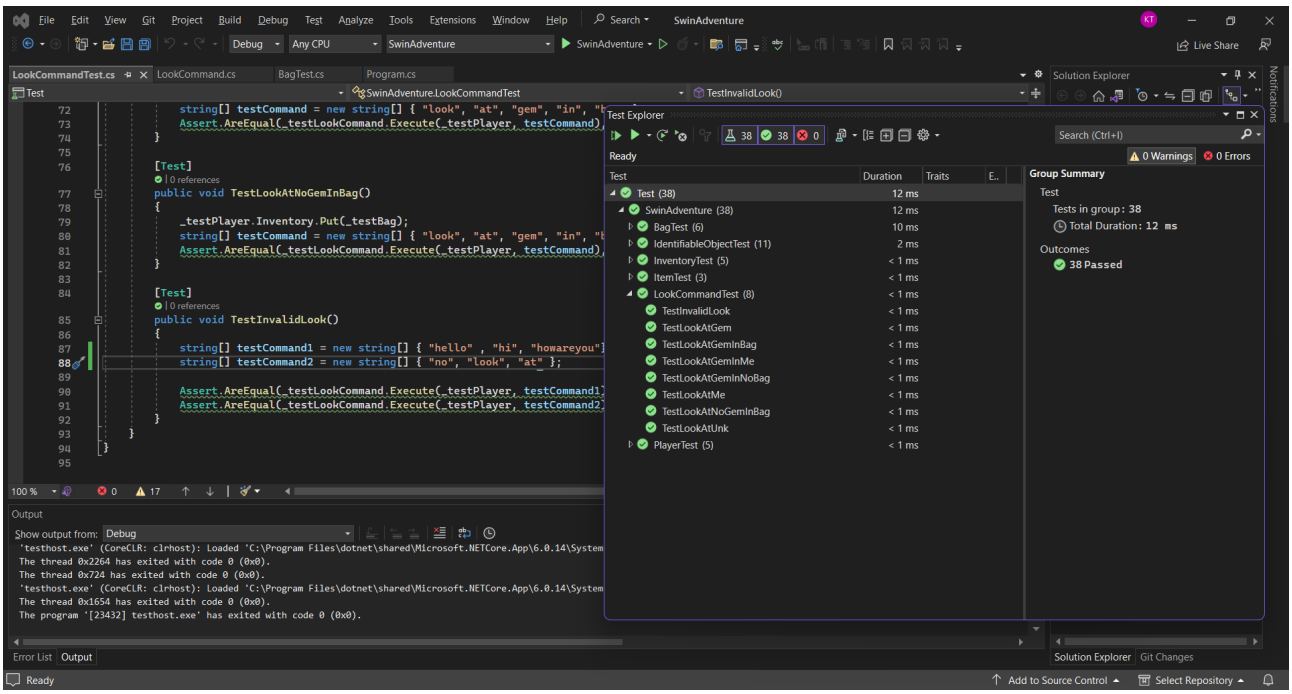
```
using System;

namespace SwinAdventure
{
    public class LookCommand : Command
    {
        public LookCommand() : base(new string[] { "look" })
        {
        }

        public override string Execute(Player p, string[] text)
        {
            if ((text.Length == 3) || (text.Length == 5))
            {
                if (text[0].ToLower() != "look")
                {
                    return "Error in look input";
                }
                if (text[1].ToLower() != "at")
                    return "What do you want to look at?";

                IHaveInventory container;
                string itemId;

                if (text.Length == 3)
                {
                    container = p as IHaveInventory;
                    itemId = text[2];
                }
                else
                {
                    if (text[3].ToLower() != "in")
                        return "What do you want to look in?";

                    container = FetchContainer(p, text[4]);
                    if (container == null)
                        return $"I cannot find the {text[4]}";
                    itemId = text[2];
                }
                return LookAtIn(itemId, container);
            }
            else
            {
                return "I don't know how to look like that";
            }
        }

        private IHaveInventory FetchContainer(Player p, string containerId)
        {
            return p.Locate(containerId) as IHaveInventory;
        }

        private string LookAtIn(string thingId, IHaveInventory container)
```

```
54          {
55              if (container.Locate(thingId) != null)
56                  return container.Locate(thingId).FullDescription;
57
58              return $"I cannot find the {thingId} in the {container.Name}";
59          }
60      }
61  }
```

```
namespace SwinAdventure
{
    public class LookCommandTest
    {
        private LookCommand _testLookCommand;
        private Player _testPlayer;
        private Bag _testBag;

        private Item _gem;

        [SetUp]
        public void SetUp()
        {
            _testLookCommand = new LookCommand();

            _gem = new Item(new string[] { "gem" }, "a gem", "This is a gem");

            _testPlayer = new Player("Trung Kien Nguyen", "I am the player");

            _testBag = new Bag(new string[] { "bag" }, "small bag", "This is a small bag");
        }

        [Test]
        public void TestLookAtMe()
        {
            string[] testCommand = new string[] { "look", "at", "inventory" };
            Assert.AreEqual(_testLookCommand.Execute(_testPlayer, testCommand), $"You are {_testPlayer.Name}, (I am the player), you are carrying:\n{_testPlayer.Inventory.ItemList}");
        }

        [Test]
        public void TestLookAtGem()
        {
            _testPlayer.Inventory.Put(_gem);

            string[] testCommand = new string[] { "look", "at", "gem" };
            Assert.AreEqual(_testLookCommand.Execute(_testPlayer, testCommand), "This is a gem");
            Assert.AreEqual(_testLookCommand.Execute(_testPlayer, testCommand), _gem.FullDescription);
        }

        [Test]
        public void TestLookAtUnk()
        {
            string[] testCommand = new string[] { "look", "at", "gem" };
            Assert.AreEqual(_testLookCommand.Execute(_testPlayer, testCommand), $"I cannot find the {testCommand[2]} in the {_testPlayer.Name}");
        }

        [Test]
```

```
48          public void TestLookAtGemInMe()
49          {
50              _testPlayer.Inventory.Put(_gem);
51
52              string[] testCommand = new string[] { "look", "at", "gem", "in",
    ↪ "inventory" };
53              Assert.AreEqual(_testLookCommand.Execute(_testPlayer, testCommand),
    ↪ "This is a gem");
54              Assert.AreEqual(_testLookCommand.Execute(_testPlayer, testCommand),
    ↪ _gem.FullDescription);
55          }
56
57          [Test]
58          public void TestLookAtGemInBag()
59          {
60              _testBag.Inventory.Put(_gem);
61              _testPlayer.Inventory.Put(_testBag);
62
63              string[] testCommand = new string[] { "look", "at", "gem", "in", "bag" };
64              Assert.AreEqual(_testLookCommand.Execute(_testPlayer, testCommand),
    ↪ "This is a gem");
65              Assert.AreEqual(_testLookCommand.Execute(_testPlayer, testCommand),
    ↪ _gem.FullDescription);
66          }
67
68          [Test]
69          public void TestLookAtGemInNoBag()
70          {
71              _testBag.Inventory.Put(_gem);
72              string[] testCommand = new string[] { "look", "at", "gem", "in", "bag" };
73              Assert.AreEqual(_testLookCommand.Execute(_testPlayer, testCommand), $"I
    ↪ cannot find the {testCommand[4]}");
74          }
75
76          [Test]
77          public void TestLookAtNoGemInBag()
78          {
79              _testPlayer.Inventory.Put(_testBag);
80              string[] testCommand = new string[] { "look", "at", "gem", "in", "bag" };
81              Assert.AreEqual(_testLookCommand.Execute(_testPlayer, testCommand), $"I
    ↪ cannot find the {testCommand[2]} in the {_testBag.Name}");
82          }
83
84          [Test]
85          public void TestInvalidLook()
86          {
87              string[] testCommand1 = new string[] { "hello" , "hi", "howareyou"};
88              string[] testCommand2 = new string[] { "no", "look", "at" };
89
90              Assert.AreEqual(_testLookCommand.Execute(_testPlayer, testCommand1),
    ↪ "Error in look input");
91              Assert.AreEqual(_testLookCommand.Execute(_testPlayer, testCommand2),
    ↪ "Error in look input");
```

```
92            }
93        }
94    }
```


File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
SwinAdventure
Debug
Any CPU
Live Share
LookCommandTest.cs
LookCommand.cs
BagTest.cs
Program.cs
Solution Explorer
Test Explorer
Ready
Test
Duration
Traits
Test (38) 12 ms
SwinAdventure (38) 12 ms
BagTest (6) 10 ms
IdentifiableObjectTest (11) 2 ms
InventoryTest (5) < 1 ms
ItemTest (3) < 1 ms
LookCommandTest (8) < 1 ms
TestInvalidLook < 1 ms
TestLookAtGem < 1 ms
TestLookAtGemInBag < 1 ms
TestLookAtGemInMe < 1 ms
TestLookAtGemInNoBag < 1 ms
TestLookAtMe < 1 ms
TestLookAtNoGemInBag < 1 ms
TestLookAtUnk < 1 ms
PlayerTest (5) < 1 ms
Group Summary
Tests in group: 38
Total Duration: 12 ms
Outcomes
38 Passed
Output
Error List
Solution Explorer
Git Changes
Add to Source Control
Select Repository