Swinburne University of Technology

COS20007 Object Oriented Programming

# Case Study - Iteration 4 - Look Command

PDF generated at 11:20 on Wednesday 11th October, 2023

```
using CaseStudy;
using System;

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

```
namespace CaseStudy
{
    public class Player : GameObject, IHaveInventory
    {
        private Inventory _inventory;
        private Location _location;

        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;
            }
            GameObject obj = _inventory.Fetch(id);
            if (obj != null)
            {
                return obj;
            }
            if (_location != null)
            {
                obj = _location.Locate(id);
                return obj;
            }
            else
            {
                return null;
            }
        }

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

        public Inventory Inventory
        {
            get
            {
                return _inventory;
            }
        }

        public Location Location
```

```
52          {
53              get
54              {
55                  return _location;
56              }
57              set
58              {
59                  _location = value;
60              }
61          }
62      }
63  }
```

```
namespace CaseStudy
{
    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);
            }
            else
            {
                return null;
            }
        }

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

        public Inventory Inventory
        {
            get
            {
                return _inventory;
            }
        }

    }
}
```

```
using System;
namespace CaseStudy
{
    public abstract class Command : IdentifiableObject
    {
        public Command(string[] ids) : base(ids)
        {
        }

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

```

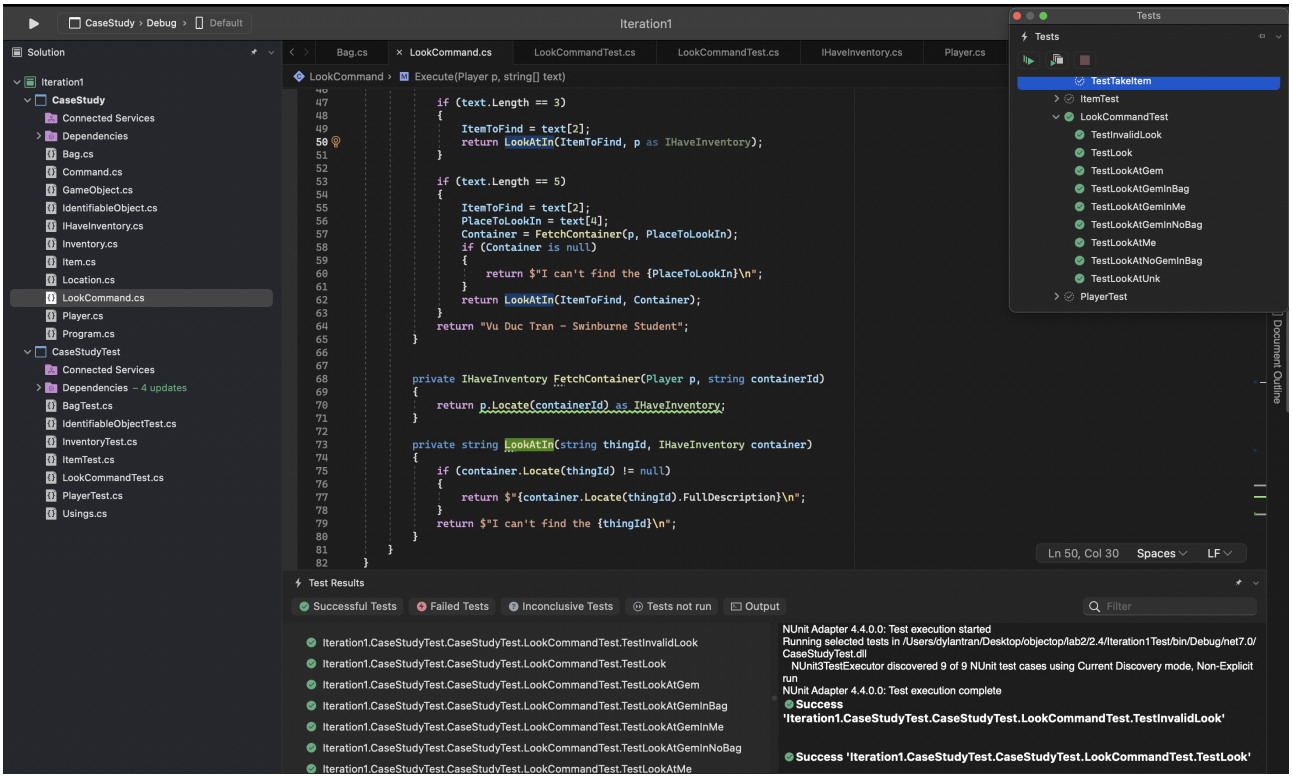
```
using System;
namespace CaseStudy
{
    public class LookCommand : Command
    {
        public LookCommand() : base(new string[] { "look" })
        {
        }

        public override string Execute(Player p, string[] text)
        {
            string ItemToFind;
            string PlaceToLookIn;
            string ItemString;
            IHaveInventory Container;

            if (text.Length == 1 && text[0] == "look")
            {
                ItemToFind = "location";
                ItemString = LookAtIn(ItemToFind, p as IHaveInventory);
                return ItemString;
            }

            if (text.Length != 3 && text.Length != 5)
            {
                return "I don't know how to look like that\n";
            }

            if (text[0] != "look")
            {
                return "Error in look input\n";
            }

            if (text[1] != "at")
            {
                return "What do you want to look at?\n";
            }

            if (text.Length == 5)
            {
                if (text[3] != "in")
                {
                    return "What do you want to look in?\n";
                }
            }

            if (text.Length == 3)
            {
                ItemToFind = text[2];
                return LookAtIn(ItemToFind, p as IHaveInventory);
            }

            if (text.Length == 5)
```

```
54              {
55                  ItemToFind = text[2];
56                  PlaceToLookIn = text[4];
57                  Container = FetchContainer(p, PlaceToLookIn);
58                  if (Container is null)
59                  {
60                      return $"I can't find the {PlaceToLookIn}\n";
61                  }
62                  return LookAtIn(ItemToFind, Container);
63              }
64              return "Vu Duc Tran - Swinburne Student";
65          }
66
67
68          private IHaveInventory FetchContainer(Player p, string containerId)
69          {
70              return p.Locate(containerId) as IHaveInventory;
71          }
72
73          private string LookAtIn(string thingId, IHaveInventory container)
74          {
75              if (container.Locate(thingId) != null)
76              {
77                  return $"{container.Locate(thingId).FullDescription}\n";
78              }
79              return $"I can't find the {thingId}\n";
80          }
81      }
82  }
83
```

```
using System;
using CaseStudy;

namespace CaseStudyTest
{
    public class LookCommandTest
    {
        private LookCommand _look;
        private Player _player1, _player2;
        private Bag _bag;
        private Location _myhouse;

        private Item _gem;
        private Item _pen;
        private string _output;

        [SetUp]
        public void Setup()
        {
            _gem = new Item(new string[] { "gem" }, "a gem", "This is a gem");
            _pen = new Item(new string[] { "pen" }, "a pen", "This is a pen");
            _look = new LookCommand();
            _player1 = new Player("Vu Duc Tran 104175614 ", "Swinburne Student");
            _player1.Inventory.Put(_bag);
            _player2 = new Player("Dylan Tran", "student");

            _bag = new Bag(new string[] { "bag" }, "Duc's bag", $"This is
↪ {_player1.FirstID} bag");
            _myhouse = new Location("My House", "My House");

            _player1.Inventory.Put(_gem);
            _bag.Inventory.Put(_gem);
            _player1.Location = _myhouse;
            _player1.Location.Items.Put(_pen);
        }

        [Test]
        public void TestLookAtMe()
        {
            _output = _look.Execute(_player1, new string[] { "look", "at",
↪ "inventory" });
            Assert.That(_output, Is.EqualTo($"You are {_player1.Name}, Swinburne
↪ Student.\nYou are carrying:\n{_player1.Inventory.ItemList}\n"), "Test Look At
↪ Me");
        }

        [Test]
        public void TestLook()
        {
            _output = _look.Execute(_player1, new string[] { "look" });
            Assert.That(_output, Is.EqualTo($"You are at: My House
↪ (location)\n\nItems at this location:\na pen (pen)\n\n"), "Test Look");
        }
```

```
49
50          [Test]
51          public void TestLookAtGem()
52          {
53              _output = _look.Execute(_player1, new string[] { "look", "at", "gem",
    ↪  "in", "inventory" });
54              Assert.That(_output, Is.EqualTo("This is a gem\n"), "Test Look At Gem");
55          }
56
57          [Test]
58          public void TestLookAtUnk()
59          {
60              _output = _look.Execute(_player1, new string[] { "look", "at", "gem2",
    ↪  "in", "inventory" });
61              Assert.That(_output, Is.EqualTo("I can't find the gem2\n"), "Test Look At
    ↪  Unk");
62          }
63
64          [Test]
65          public void TestLookAtGemInMe()
66          {
67              _output = _look.Execute(_player1, new string[] { "look", "at", "gem",
    ↪  "in", "me" });
68              Assert.That(_output, Is.EqualTo("This is a gem\n"), "Test Look At Gem in
    ↪  Me");
69          }
70
71          [Test]
72          public void TestLookAtGemInBag()
73          {
74              _output = _look.Execute(_player1, new string[] { "look", "at", "gem",
    ↪  "in", "bag" });
75              Assert.That(_output, Is.EqualTo("This is a gem\n"), "Test Look At Gem in
    ↪  Bag");
76          }
77
78          [Test]
79          public void TestLookAtGemInNoBag()
80          {
81              _output = _look.Execute(_player2, new string[] { "look", "at", "gem",
    ↪  "in", "bag" });
82              Assert.That(_output, Is.EqualTo("I can't find the bag\n"), "Test Look At
    ↪  Gem in No Bag");
83          }
84
85          [Test]
86          public void TestLookAtNoGemInBag()
87          {
88              _output = _look.Execute(_player1, new string[] { "look", "at", "gem2",
    ↪  "in", "bag" });
89              Assert.That(_output, Is.EqualTo("I can't find the gem2\n"), "Test Look At
    ↪  No Gem in Bag");
90          }
```

```
91
92          [Test]
93          public void TestInvalidLook()
94          {
95              _output = _look.Execute(_player1, new string[] { "look", "around" });
96              Assert.That(_output, Is.EqualTo("I don't know how to look like that\n"),
    ↪  "Test Invalid Look");
97
98              _output = _look.Execute(_player1, new string[] { "hello", "hello",
    ↪  "hello" });
99              Assert.That(_output, Is.EqualTo("Error in look input\n"), "Test Invalid
    ↪  Look");
100
101             _output = _look.Execute(_player1, new string[] { "look", "in", "a", "in",
    ↪  "b" });
102             Assert.That(_output, Is.EqualTo("What do you want to look at?\n"), "Test
    ↪  Invalid Look");
103
104             _output = _look.Execute(_player1, new string[] { "look", "at", "a", "at",
    ↪  "b" });
105             Assert.That(_output, Is.EqualTo("What do you want to look in?\n"), "Test
    ↪  Invalid Look");
106         }
107     }
108 }
109
```


Iteration1
Solution
Iteration1
CaseStudy
Connected Services
Dependencies
Bag.cs
Command.cs
GameObject.cs
IdentifiableObject.cs
IHaveInventory.cs
Inventory.cs
Item.cs
Location.cs
LookCommand.cs
Player.cs
Program.cs
CaseStudyTest
Connected Services
Dependencies – 4 updates
BagTest.cs
IdentifiableObjectTest.cs
InventoryTest.cs
ItemTest.cs
LookCommandTest.cs
PlayerTest.cs
Usings.cs
Bag.cs
LookCommand.cs
LookCommandTest.cs
LookCommandTest.cs
IHaveInventory.cs
Player.cs
LookCommand › Execute(Player p, string[] text)
Tests
TestTakeItem
ItemTest
LookCommandTest
TestInvalidLook
TestLook
TestLookAtGem
TestLookAtGemInBag
TestLookAtGemInMe
TestLookAtGemInNoBag
TestLookAtMe
TestLookAtNoGemInBag
TestLookAtUnk
PlayerTest
Ln 50, Col 30
Spaces
LF
Test Results
Successful Tests
Failed Tests
Inconclusive Tests
Tests not run
Output
Filter
Iteration1.CaseStudyTest.CaseStudyTest.LookCommandTest.TestInvalidLook
Iteration1.CaseStudyTest.CaseStudyTest.LookCommandTest.TestLook
Iteration1.CaseStudyTest.CaseStudyTest.LookCommandTest.TestLookAtGem
Iteration1.CaseStudyTest.CaseStudyTest.LookCommandTest.TestLookAtGemInBag
Iteration1.CaseStudyTest.CaseStudyTest.LookCommandTest.TestLookAtGemInMe
Iteration1.CaseStudyTest.CaseStudyTest.LookCommandTest.TestLookAtGemInNoBag
Iteration1.CaseStudyTest.CaseStudyTest.LookCommandTest.TestLookAtMe
NUnit Adapter 4.4.0.0: Test execution started
Running selected tests in /Users/dylantran/Desktop/objectop/lab2/2.4/Iteration1Test/bin/Debug/net7.0/CaseStudyTest.dll
NUnit3TestExecutor discovered 9 of 9 NUnit test cases using Current Discovery mode, Non-Explicit run
NUnit Adapter 4.4.0.0: Test execution complete
Success 'Iteration1.CaseStudyTest.CaseStudyTest.LookCommandTest.TestInvalidLook'
Success 'Iteration1.CaseStudyTest.CaseStudyTest.LookCommandTest.TestLook'