Swinburne University of Technology

Object Oriented Programming (2022 S1)

Doubtfire Submission

Task 5.2P: Case Study Iteration 4: Look Command

Submitted By:
Vaissheenavi Prabakaran
103508183
2022/05/13 22:54

Tutor:
Jai Cornes

May 13, 2022

```
using System;
using System.Collections.Generic;

namespace Task_5._2
{
    public class IdentifiableObject
    {
        private List<string> _identifiers = new List<string>();



        public IdentifiableObject(string[] idents)
        {
            foreach (string id in idents)
            {
                _identifiers.Add(id.ToLower());
            }
        }

        //private List<string> _identifiers= new List<string>();




        public bool AreYou(string id)
        {
            //return _identifiers.Contains(id.ToLower());

            foreach (string idAY in _identifiers)
            {
                if (id.ToLower() == idAY)
                {
                    return true;
                }

                //return false;
            }

            return false;
        }


        public string FirstID
        {
            get
            {
                if (_identifiers.Count > 0)
                {
                    return _identifiers[0];
                }

                return "";
            }
```

```
54          }
55
56          public void AddIdentifier(string id)
57          {
58              //id = id.ToLower();
59              _identifiers.Add(id.ToLower());
60
61              return;
62          }
63      }
64  }
```

```
using System;

namespace Task_5._2
{
    public abstract class GameObject : IdentifiableObject
    {
        private string _description;
        private string _name;




        public string Name
        {
            get
            {
                return _name;
            }

        }

        public string ShortDescription
        {
            get
            {
                return _name + " (" + FirstID + ")";
            }

        }


        public virtual string FullDescription
        {
            get
            {
                return _description;
            }

        }


        public GameObject(string[] ids, string name, string desc) : base(ids)
        {
            _name = name;
            _description = desc;
        }


    }
}
```

```
using System;
namespace Task_5._2
{
    public class Player : GameObject, IHaveInventory
    {
        //already in GameObject
        //private string _description;
        //private string _name;
        private Inventory inventory;


        public Player(string name, string desc) : base(new string[] { "myself",
        ↪  "inventory" }, 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 inventory.ItemList;
                return "You are carrying: " + Name + inventory.ItemList;

                //return inventorydescription;
            }
        }

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

```
using System;
using System.Collections.Generic;
namespace Task_5._2
{
    public class Inventory
    {
        private List<Item> _items = new List<Item>();


        public Inventory()
        {
            //foreach (Item i in _items)
            //{
            //if (i.AreYou(id))
            //{
            //return true;
            //}

            //else
            //{
            //return false;
            //}

            //}


            //_items = new List<Item>();

        }

        public bool HasItem(string id)
        {
            foreach (Item i in _items)
            {
                if (i.AreYou(id))
                {
                    return true;
                }


            }
            return false;

        }



        public void Put(Item itm)
        {
            _items.Add(itm);
        }


```

```
54          public Item Fetch(string id)
55          {
56              foreach (Item i in _items)
57              {
58                  if (i.AreYou(id))
59                  {
60                      Item itemToFetch = i;
61
62                      //return true;
63                      return itemToFetch;
64                  }
65
66                  //return null;
67
68              }
69
70              return null;
71          }
72
73
74          public Item Take(string id)
75          {
76              Item i = Fetch(id);
77              if (i != null)
78              {
79                  _items.Remove(i);
80                  return i;
81              }
82
83              return null;
84          }
85
86
87          public string ItemList
88          {
89              get
90              {
91                  string iList = "";
92                  foreach (Item i in _items)
93                  {
94                      iList += "\t" + i.ShortDescription + "\n";
95                  }
96
97                  if (iList == null)
98                  {
99                      return "Item not found!";
100                 }
101
102                 return iList;
103             }
104
105         }
106     }
```

```
107 }
```

```
using System;
namespace Task_5._2
{
    public class Item : GameObject
    {
        //already in GameObject
        //private string _description;
        //private string _name;


        public Item(string[] idents, string name, string desc) : base(idents, name,
        ↪ desc)
        {
            //_name = name;
            //_description = desc;
        }
    }
}
```

```
using System;
using System.Collections.Generic;
namespace Task_5._2
{
    public class Bag : Item, IHaveInventory
    {
        private Inventory _inventory;

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

        public Inventory Inventory
        {
            get
            {
                return _inventory;
            }
        }

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

        }

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

            else if (_inventory.HasItem(id))
            {
                return _inventory.Fetch(id);
            }

            return null;
        }

    }
}
```

```
using System;
namespace Task_5._2
{
    public interface IHaveInventory
    {
        GameObject Locate(string id);

        public string Name
        {
            get;
        }


    }
}
```

```
using System;
namespace Task_5._2
{
    public abstract class Command : IdentifiableObject
    {
        public Command(string [] ids) : base(ids)
        {
        }

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

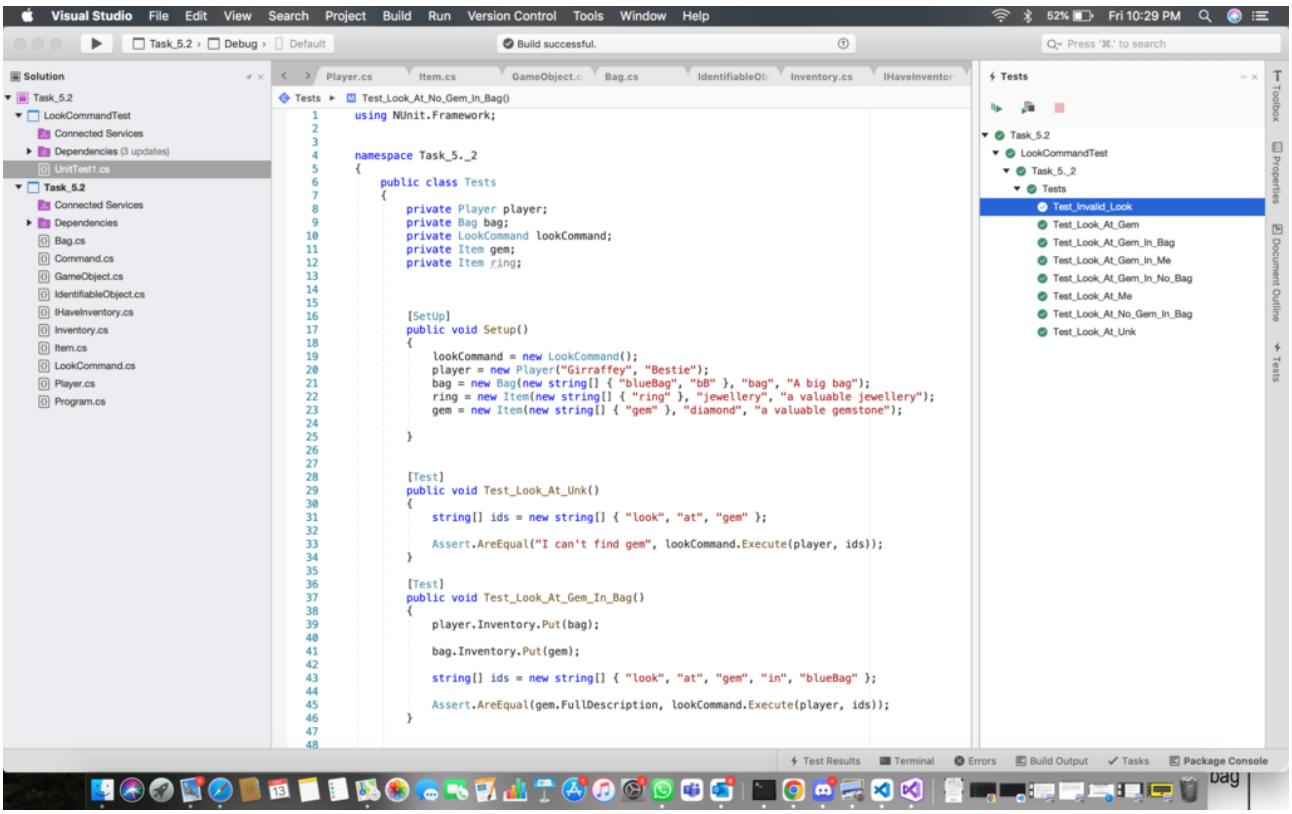
```
using System;
namespace Task_5._2
{
    public class LookCommand : Command
    {
        public LookCommand() : base (new string[] {"look"})
        {
        }

        public override string Execute(Player p, string[] text)
        {
            if(text.Length == 3 || text.Length == 5)
            {
                if (text[0] == "look")
                {
                    if (text[1] == "at")
                    {
                        if(text.Length == 3)
                        {
                            return LookAtIn(text[2], p);
                        }

                        else
                        {
                            if (text[3] == "in")
                            {
                                if(FetchContainer(p, text[4]) == null)
                                {
                                    return "I can't find " + text[2] + " in " +
                                    ↪ text[4];
                                }

                                else
                                {
                                    return LookAtIn(text[2], FetchContainer(p,
                                    ↪ text[4]));
                                }
                            }

                            else
                            {
                                return "What do you want to look in?";
                            }
                        }
                    }

                    else
                    {
                        return "What do you want to look at?";
                    }
                }

                else
```

```
52                  {
53                      return "Error look input";
54                  }
55              }
56
57              else
58              {
59                  return "I don't know how to look like that";
60              }
61          }
62
63
64          private string LookAtIn(string ThingID, IHaveInventory container)
65          {
66              if(container.Locate(ThingID) == null)
67              {
68                  return "I can't find " + ThingID;
69              }
70
71              else
72              {
73                  return container.Locate(ThingID).FullDescription;
74              }
75          }
76
77          private IHaveInventory FetchContainer(Player p, string ContainerID)
78          {
79              return p.Locate(ContainerID) as IHaveInventory;
80          }
81      }
82  }
```

```
using NUnit.Framework;


namespace Task_5._2
{
    public class Tests
    {
        private Player player;
        private Bag bag;
        private LookCommand lookCommand;
        private Item gem;
        private Item ring;



        [SetUp]
        public void Setup()
        {
            lookCommand = new LookCommand();
            player = new Player("Girraffey", "Bestie");
            bag = new Bag(new string[] { "blueBag", "bB" }, "bag", "A big bag");
            ring = new Item(new string[] { "ring" }, "jewellery", "a valuable
            ↪  jewellery");
            gem = new Item(new string[] { "gem" }, "diamond", "a valuable
            ↪  gemstone");

        }


        [Test]
        public void Test_Look_At_Unk()
        {
            string[] ids = new string[] { "look", "at", "gem" };

            Assert.AreEqual("I can't find gem", lookCommand.Execute(player, ids));
        }

        [Test]
        public void Test_Look_At_Gem_In_Bag()
        {
            player.Inventory.Put(bag);

            bag.Inventory.Put(gem);

            string[] ids = new string[] { "look", "at", "gem", "in", "blueBag" };

            Assert.AreEqual(gem.FullDescription, lookCommand.Execute(player, ids));
        }


        [Test]
        public void Test_Look_At_Me()
        {
```

```
52              string[] ids = new string[] { "look", "at", "inventory" };
53
54              Assert.AreEqual(player.FullDescription, lookCommand.Execute(player,
                ↪  ids));
55          }
56
57          [Test]
58          public void Test_Look_At_Gem()
59          {
60              player.Inventory.Put(gem);
61
62              string[] ids = new string[] { "look", "at", "gem" };
63
64              Assert.AreEqual(gem.FullDescription, lookCommand.Execute(player, ids));
65          }
66
67
68
69          [Test]
70          public void Test_Look_At_Gem_In_Me()
71          {
72              player.Inventory.Put(gem);
73
74              string[] ids = new string[] { "look", "at", "gem", "in", "inventory" };
75
76              Assert.AreEqual(gem.FullDescription, lookCommand.Execute(player, ids));
77          }
78
79          [Test]
80          public void Test_Look_At_No_Gem_In_Bag()
81          {
82              bag = new Bag(new string[] { "blueBag", "bB" }, "bag", "A big bag");
83
84              player.Inventory.Put(bag);
85
86              player.Inventory.Put(gem);
87
88              string[] ids = new string[] { "look", "at", "gem", "in", "bag" };
89
90              Assert.AreEqual("I can't find gem in bag", lookCommand.Execute(player,
                ↪  ids));
91          }
92
93
94          [Test]
95          public void Test_Look_At_Gem_In_No_Bag()
96          {
97              bag = new Bag(new string[] { "blueBag", "bB" }, "bag", "A big bag");
98
99              string[] ids = new string[] { "look", "at", "bag" };
100
101             Assert.AreEqual("I can't find bag",lookCommand.Execute(player, ids));
102         }
```

```
103
104
105
106         [Test]
107         public void Test_Invalid_Look()
108         {
109             Assert.AreEqual("Error look input" , lookCommand.Execute(player, new
                ↪  string[] { "pick", "up", "ring" }));
110             Assert.AreEqual("I don't know how to look like that" ,
                ↪  lookCommand.Execute(player, new string[] { "look", "at"}));
111             Assert.AreEqual("What do you want to look in?",
                ↪  lookCommand.Execute(player, new string[] { "look", "at", "table",
                ↪  "and", "chair" }));
112             Assert.AreEqual("What do you want to look at?",
                ↪  lookCommand.Execute(player, new string[] { "look", "test", "gem"
                ↪  }));
113         }
114
115     }
116
117 }
```


Visual Studio File Edit View Search Project Build Run Version Control Tools Window Help
Build successful.
Solution
Task_5.2
LookCommandTest
Connected Services
Dependencies (3 updates)
UnitTest1.cs
Task_5.2
Connected Services
Dependencies
Bag.cs
Command.cs
GameObject.cs
IIdentifiableObject.cs
IHaveInventory.cs
Inventory.cs
Item.cs
LookCommand.cs
Player.cs
Program.cs
Player.cs
Item.cs
GameObject.c
Bag.cs
Inventory.cs
Tests
Test_Look_At_No_Gem_In_Bag()
using NUnit.Framework;
namespace Task_5._2
{
public class Tests
{
private Player player;
private Bag bag;
private LookCommand lookCommand;
private Item gem;
private Item ring;
[SetUp]
public void Setup()
{
lookCommand = new LookCommand();
player = new Player("Girraffey", "Bestie");
bag = new Bag(new string[] { "blueBag", "bB" }, "bag", "A big bag");
ring = new Item(new string[] { "ring" }, "jewellery", "a valuable jewellery");
gem = new Item(new string[] { "gem" }, "diamond", "a valuable gemstone");
}
[Test]
public void Test_Look_At_Unk()
{
string[] ids = new string[] { "look", "at", "gem" };
Assert.AreEqual("I can't find gem", lookCommand.Execute(player, ids));
}
[Test]
public void Test_Look_At_Gem_In_Bag()
{
player.Inventory.Put(bag);
bag.Inventory.Put(gem);
string[] ids = new string[] { "look", "at", "gem", "in", "blueBag" };
Assert.AreEqual(gem.FullDescription, lookCommand.Execute(player, ids));
}
Tests
Task_5.2
LookCommandTest
Task_5._2
Tests
Test_Invalid_Look
Test_Look_At_Gem
Test_Look_At_Gem_In_Bag
Test_Look_At_Gem_In_Me
Test_Look_At_Gem_In_No_Bag
Test_Look_At_Me
Test_Look_At_No_Gem_In_Bag
Test_Look_At_Unk
Test Results
Terminal
Errors
Build Output
Tasks
Package Console