Swinburne University of Technology

COS20007 Object Oriented Programming

Case Study - Iteration 2 - Players Items and Inventory

PDF generated at 15:18 on Monday 25th September, 2023

Produced by OOP Submission Formatter

```
using System;
using Iteration2;

namespace Iteration2
{
    public abstract class Game_Object : Identifiable_object
    {
        private string _description;
        private string _name;
        public Game_Object(string[] ids, string name, string desc) : base(ids)
        {
            _name = name;
            _description = desc;
        }
        public string Name //readonly property
        {
            get
            {
                return _name;
            }
        }
        public string ShortDescription //readonly property returns the required
    ↪  format of string displaying items from the list with required description
        {
            get
            {
                return "a " + _name + " " + "(" + FirstId + ")";
            }
        }
        public virtual string FullDescription //readonly returns the description.
        {
            get
            {
                return _description;
            }
        }
    }
}

```

```
using System;
namespace Iteration2
{
    public class Player: Game_Object
    {
        Inventory _inventory = new Inventory();

        public Player(string name, string desc):base(new string[] {"me",
↪   "inventory"}, name, desc)
        {
            //Inventory _inventory = new() Inventory;
        }

        public Game_Object Locate(string id)  //Locate method returns the item using
↪   id and calls the Fetch method to retrieve the item.
        {
            if(AreYou(id) == true)
            {
                return this;
            }
            return _inventory.Fetch(id);
        }
        public Inventory Inv  //readonly method for Inventory
        {
            get
            {
                return _inventory;
            }
        }
        public override string FullDescription   // Full description returns the
↪   name, full description and itemlist.
        {
            get
            {
                return ("You are " + Name + " " + base.FullDescription + "." + "\nYou
↪   are carrying\n" + Inv.ItemList);
            }
        }
    }
}

```

```
using System;
using System.Xml.Linq;

namespace Iteration2
{
    public class PlayerTest
    {

        Item shovel, sword, computer;
        Player pl;



        [SetUp()]
        public void Constructor_PlayerTest()
        {

            pl = new("Fred", "the mighty programmer");
            shovel = new(new string[] { "shovel" }, "shovel", "");
            sword = new(new string[] { "sword" }, "sword", "bronze");
            computer = new(new string[] { "pc" }, "computer", "small");


            pl.Inv.Put(shovel);
            pl.Inv.Put(sword);
            pl.Inv.Put(computer);


        }
        [Test()]
        public void Test_Player_is_Identifiable()
        {
            Assert.IsTrue(pl.AreYou("me"));
        }
        [Test()]
        public void Test_Player_Locates_Items()
        {
            Assert.AreEqual(pl.Locate("shovel"),shovel);
            Assert.IsTrue(pl.Inv.HasItem("shovel"));


            Assert.AreEqual(pl.Locate("sword"),sword);
            Assert.IsTrue(pl.Inv.HasItem("sword"));


            Assert.AreEqual(pl.Locate("pc"),computer);
            Assert.IsTrue(pl.Inv.HasItem("pc"));

        }
        [Test()]
        public void Test_Player_Locates_Itself()
        {
            Assert.AreEqual(pl.Locate("me"),pl);
```

```
54              Assert.AreEqual(pl.Locate("inventory"), pl);
55
56          }
57          [Test()]
58          public void Test_Player_Locates_Nothing()
59          {
60              Assert.AreEqual(pl.Locate("food"), null);
61
62              Assert.AreEqual(pl.Locate("boat"), null);
63          }
64          [Test()]
65          public void Test_Player_full_Description()
66          {
67              Assert.AreEqual("You are Fred the mighty programmer.\n" + "You are
    ↪  carrying\n" + "\ta shovel (shovel)\n\ta sword (sword)\n\ta computer (pc)\n",
    ↪  pl.FullDescription);
68          }
69      }
70  }
71
```

```
using System;
namespace Iteration2
{
    public class Item:Game_Object
    {
        public Item(string[] idents, string name, string desc):base(idents, name,
↪   desc)
        {
        }
    }
}

```

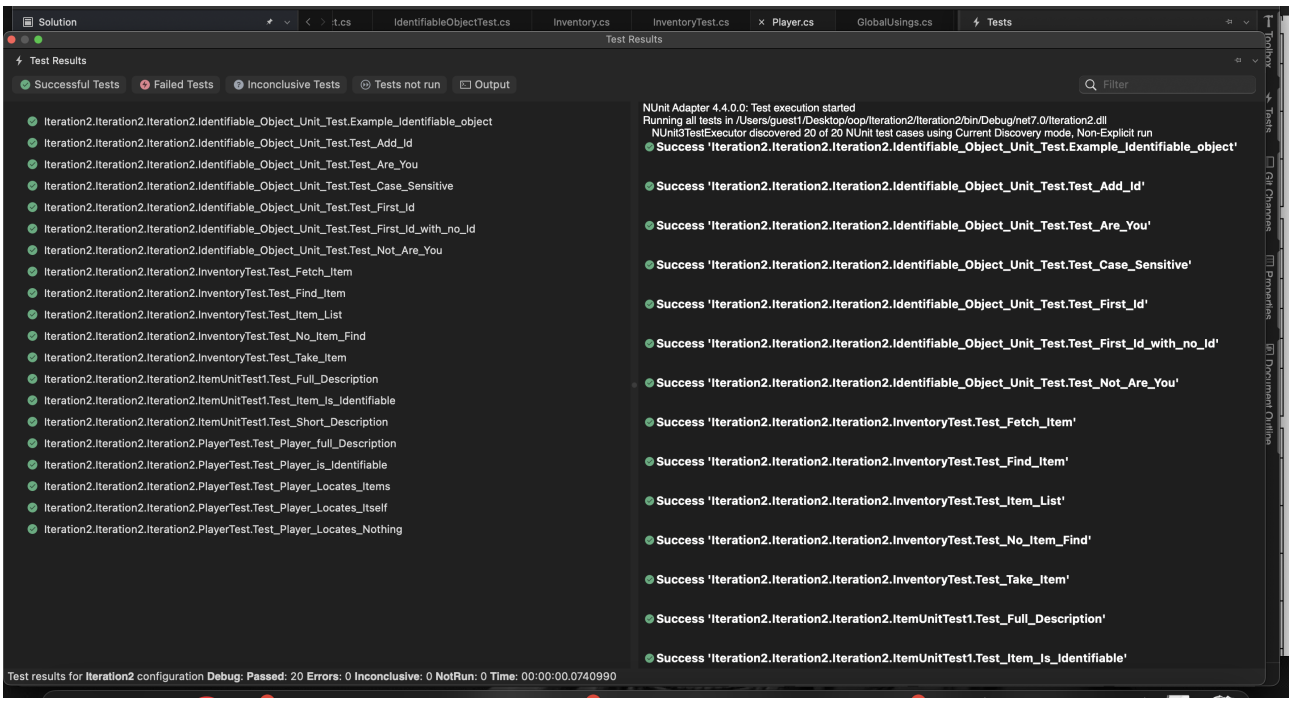
```
using Iteration2;

namespace Iteration2;


    [TestFixture()]
    public class ItemUnitTest1
    {
        Item itemTest;


        [SetUp()]
        public void Setup()
        {
            itemTest = new(new string[] { "pc" }, "computer", "small");

        }

        [Test()]
        public void Test_Item_Is_Identifiable()
        {
            Assert.IsTrue(itemTest.AreYou("pc"));
        }

        [Test()]
        public void Test_Short_Description()
        {
            Assert.AreEqual(itemTest.ShortDescription,"a computer (pc)");
        }

        [Test()]
        public void Test_Full_Description()
        {
            Assert.AreEqual(itemTest.FullDescription, "small");
        }
    }

```

```
using System;
namespace Iteration2
{
    public class Inventory
    {
        private List<Item> _items = new List<Item>();  //initialising a new list of items.
        public Inventory()
        {
        }

        public bool HasItem(string id)  //Method1 to check if the item i is in list
        {
            foreach(Item i in _items)
            {
                if (i.AreYou(id))
                {
                    return true;  //return true if item in list
                }

            }
            return false;  //otherwise always return false
        }
        public void Put(Item itm)  //Method to add the item into the list.
        {
            _items.Add(itm);
        }
        public Item Take(string id)  // Method to remove item from list
        {
            Item i = Fetch(id);  // first retrieve the item then remove it if it is not null.

            if (_items != null)
            {
                _items.Remove(i);  // remove the item from the list
                return i;// return the item after removing
            }
            return null;
        }
        public Item Fetch(string id)  //method to retrieve item using id
        {
            foreach(Item i in _items)
            {
                if (i.AreYou(id))
                {
                    return i;  //return item from list with corresponding id
                }
            }
            return null;

        }
        public string ItemList  //method to return list in format with description and firstid.
```

```
51          {
52              get
53              {
54                  string l = "";
55
56                  foreach (Item i in _items)
57                  {
58                      l += "\t" + i.ShortDescription + "\n";
59                  }
60                  return l;
61
62              }
63          }
64      }
65  }
66
```

```
using Iteration2;

namespace Iteration2;

[TestFixture()]
public class InventoryTest
{
    Inventory inv;
    Item shovel, sword, computer;



    [SetUp()]
    public void Setup()
    {
        inv = new();
        shovel = new(new string[] { "shovel" }, "shovel", "");
        inv.Put(shovel);

        sword = new(new string[] { "sword" }, "sword", "bronze");
        inv.Put(sword);

        computer = new(new string[] { "pc" }, "computer", "small");
        inv.Put(computer);


    }
    [Test()]
    public void Test_Find_Item()
    {

        Assert.IsTrue(inv.HasItem("shovel"));
        Assert.IsTrue(inv.HasItem("sword"));
        Assert.IsTrue(inv.HasItem("pc"));
    }

    [Test()]
    public void Test_No_Item_Find()
    {
        Assert.IsFalse(inv.HasItem("food"));

        Assert.IsFalse(inv.HasItem("boat"));
    }

    [Test()]
    public void Test_Fetch_Item()
    {
        Assert.AreEqual(inv.Fetch("shovel"), shovel);
        Assert.IsTrue(inv.HasItem("shovel"));

        Assert.AreEqual(inv.Fetch("sword"),sword);
        Assert.IsTrue(inv.HasItem("sword"));

```

```
54          Assert.AreEqual(inv.Fetch("pc"),computer);
55          Assert.IsTrue(inv.HasItem("pc"));
56      }
57
58      [Test()]
59      public void Test_Take_Item()
60      {
61          inv.Take("shovel");
62          Assert.IsFalse(inv.HasItem("shovel"));
63      }
64
65      [Test()]
66      public void Test_Item_List()
67      {
68          Assert.AreEqual(("\ta shovel (shovel)\n\ta sword (sword)\n\ta computer
    ↪ (pc)\n"), (inv.ItemList));
69      }
70
71
72  }
```


Solution
IdentifiableObjectTest.cs
Inventory.cs
InventoryTest.cs
Player.cs
GlobalUsings.cs
Tests
Test Results
Test Results
Successful Tests
Failed Tests
Inconclusive Tests
Tests not run
Output
Filter
Iteration2.Iteration2.Iteration2.Identifiable_Object_Unit_Test.Example_Identifiable_object
Iteration2.Iteration2.Iteration2.Identifiable_Object_Unit_Test.Test_Add_Id
Iteration2.Iteration2.Iteration2.Identifiable_Object_Unit_Test.Test_Are_You
Iteration2.Iteration2.Iteration2.Identifiable_Object_Unit_Test.Test_Case_Sensitive
Iteration2.Iteration2.Iteration2.Identifiable_Object_Unit_Test.Test_First_Id
Iteration2.Iteration2.Iteration2.Identifiable_Object_Unit_Test.Test_First_Id_with_no_Id
Iteration2.Iteration2.Iteration2.Identifiable_Object_Unit_Test.Test_Not_Are_You
Iteration2.Iteration2.Iteration2.InventoryTest.Test_Fetch_Item
Iteration2.Iteration2.Iteration2.InventoryTest.Test_Find_Item
Iteration2.Iteration2.Iteration2.InventoryTest.Test_Item_List
Iteration2.Iteration2.Iteration2.InventoryTest.Test_No_Item_Find
Iteration2.Iteration2.Iteration2.InventoryTest.Test_Take_Item
Iteration2.Iteration2.Iteration2.ItemUnitTest1.Test_Full_Description
Iteration2.Iteration2.Iteration2.ItemUnitTest1.Test_Item_Is_Identifiable
Iteration2.Iteration2.Iteration2.ItemUnitTest1.Test_Short_Description
Iteration2.Iteration2.Iteration2.PlayerTest.Test_Player_full_Description
Iteration2.Iteration2.Iteration2.PlayerTest.Test_Player_is_Identifiable
Iteration2.Iteration2.Iteration2.PlayerTest.Test_Player_Locates_Items
Iteration2.Iteration2.Iteration2.PlayerTest.Test_Player_Locates_Itself
Iteration2.Iteration2.Iteration2.PlayerTest.Test_Player_Locates_Nothing
NUnit Adapter 4.4.0.0: Test execution started
Running all tests in /Users/guest1/Desktop/oop/Iteration2/Iteration2/bin/Debug/net7.0/Iteration2.dll
NUnit3TestExecutor discovered 20 of 20 NUnit test cases using Current Discovery mode, Non-Explicit run
Success 'Iteration2.Iteration2.Iteration2.Identifiable_Object_Unit_Test.Example_Identifiable_object'
Success 'Iteration2.Iteration2.Iteration2.Identifiable_Object_Unit_Test.Test_Add_Id'
Success 'Iteration2.Iteration2.Iteration2.Identifiable_Object_Unit_Test.Test_Are_You'
Success 'Iteration2.Iteration2.Iteration2.Identifiable_Object_Unit_Test.Test_Case_Sensitive'
Success 'Iteration2.Iteration2.Iteration2.Identifiable_Object_Unit_Test.Test_First_Id'
Success 'Iteration2.Iteration2.Iteration2.Identifiable_Object_Unit_Test.Test_First_Id_with_no_Id'
Success 'Iteration2.Iteration2.Iteration2.Identifiable_Object_Unit_Test.Test_Not_Are_You'
Success 'Iteration2.Iteration2.Iteration2.InventoryTest.Test_Fetch_Item'
Success 'Iteration2.Iteration2.Iteration2.InventoryTest.Test_Find_Item'
Success 'Iteration2.Iteration2.Iteration2.InventoryTest.Test_Item_List'
Success 'Iteration2.Iteration2.Iteration2.InventoryTest.Test_No_Item_Find'
Success 'Iteration2.Iteration2.Iteration2.InventoryTest.Test_Take_Item'
Success 'Iteration2.Iteration2.Iteration2.ItemUnitTest1.Test_Full_Description'
Success 'Iteration2.Iteration2.Iteration2.ItemUnitTest1.Test_Item_Is_Identifiable'
Test results for Iteration2 configuration Debug: Passed: 20 Errors: 0 Inconclusive: 0 NotRun: 0 Time: 00:00:00.0740990
Toolbox
Tests
Git Changes
Properties
Document Outline