Swinburne University of Technology

Object Oriented Programming (2022 S1)

Doubtfire Submission

# Task 5.1P: Case Study Iteration 3: Bags

*Submitted By:*
Vaissheenavi Prabakaran
103508183
2022/04/25 23:06

*Tutor:*
Jai Cornes

April 26, 2022

Produced by Doubtfire

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

namespace Task_5._1
{
    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 "";
            }
        }

        public void AddIdentifier(string id)
        {
            //id = id.ToLower();
            _identifiers.Add(id.ToLower());

            return;
        }
    }
}
```

```
using System;

namespace Task_5._1
{
    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._1
{
    public class Player : GameObject
    {
        //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;
                //string inventorydescription = "You're carrying: " + Inventory.ItemList;
                //return inventorydescription;
            }
        }

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

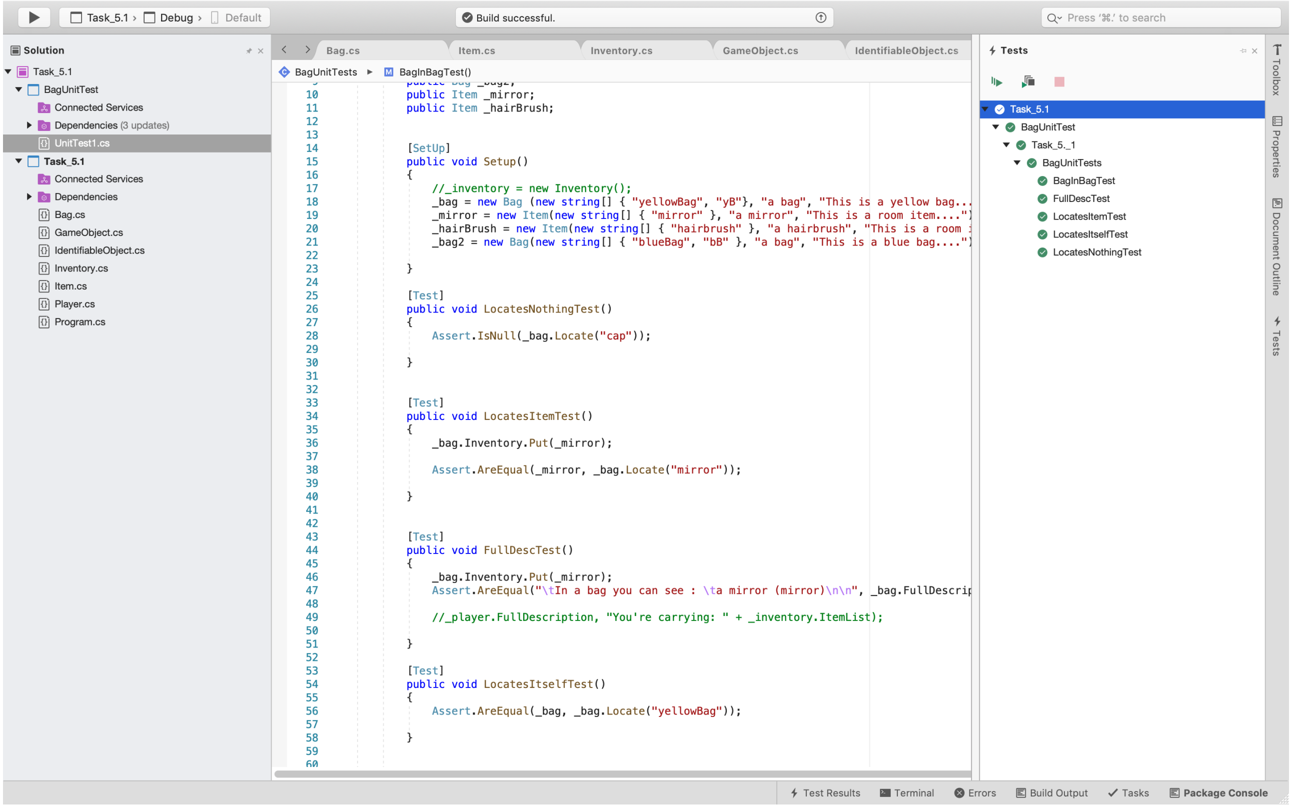
```
using System;
using System.Collections.Generic;
namespace Task_5._1
{
    public class Inventory
    {
        private List<Item> _items = new List<Item>();


        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);
        }


        public Item Fetch(string id)
        {
            foreach (Item i in _items)
            {
                if (i.AreYou(id))
                {
                    Item itemToFetch = i;

                    //return true;
                    return itemToFetch;
                }

                //return null;

            }

            return null;
        }


        public Item Take(string id)
        {
            Item i = Fetch(id);
            if (i != null)
            {
                _items.Remove(i);
                return i;
            }

            return null;
        }


        public string ItemList
        {
            get
            {
                string iList = "";
                foreach (Item i in _items)
                {
                    iList += "\t" + i.ShortDescription + "\n";
                }

                if (iList == null)
                {
                    return "Item not found!";
                }

                return iList;
            }

        }
    }
```

```
}
```

```
using System;
namespace Task_5._1
{
    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._1
{
    public class Bag : Item
    {
        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 NUnit.Framework;

namespace Task_5._1
{
    public class BagUnitTests
    {

        public Bag _bag;
        public Bag _bag2;
        public Item _mirror;
        public Item _hairBrush;


        [SetUp]
        public void Setup()
        {
            //_inventory = new Inventory();
            _bag = new Bag (new string[] { "yellowBag", "yB"}, "a bag", "This is a yellow bag....");
            _mirror = new Item(new string[] { "mirror" }, "a mirror", "This is a room item....");
            _hairBrush = new Item(new string[] { "hairbrush" }, "a hairbrush", "This is a room item....");
            _bag2 = new Bag(new string[] { "blueBag", "bB" }, "a bag", "This is a blue bag....");

        }

        [Test]
        public void LocatesNothingTest()
        {
            Assert.IsNull(_bag.Locate("cap"));

        }


        [Test]
        public void LocatesItemTest()
        {
            _bag.Inventory.Put(_mirror);

            Assert.AreEqual(_mirror, _bag.Locate("mirror"));

        }


        [Test]
        public void FullDescTest()
        {
            _bag.Inventory.Put(_mirror);
            Assert.AreEqual("\tIn a bag you can see : \ta mirror (mirror)\n\n", _bag.FullDescription);

            //_player.FullDescription, "You're carrying: " + _inventory.ItemList);

        }

        [Test]
        public void LocatesItselfTest()
        {
            Assert.AreEqual(_bag, _bag.Locate("yellowBag"));

        }


        [Test]
        public void BagInBagTest()
        {
            //Bag firstbag = _bag = new Bag(new string[] { "yellowBag", "yB" }, "a bag", "This is a yellow bag....");
            //Bag secondbag = new Bag(new string[] { "blueBag", "bB" }, "a bag", "This is a blue bag....");

            _bag.Inventory.Put(_bag2);

            Assert.AreEqual(_bag2, _bag.Locate("blueBag"));

            _bag.Inventory.Put(_hairBrush);

            Assert.AreEqual(_hairBrush, _bag.Locate("hairbrush"));

            _bag2.Inventory.Put(_mirror);

            Assert.AreNotEqual(_mirror, _bag.Locate("_mirror"));
        }
    }
}
```



Build successful.

```
        public Bag _bag2;
        public Item _mirror;
        public Item _hairBrush;


        [SetUp]
        public void Setup()
        {
            //_inventory = new Inventory();
            _bag = new Bag (new string[] { "yellowBag", "yB"}, "a bag", "This is a yellow bag...
            _mirror = new Item(new string[] { "mirror" }, "a mirror", "This is a room item....")
            _hairBrush = new Item(new string[] { "hairbrush" }, "a hairbrush", "This is a room i
            _bag2 = new Bag(new string[] { "blueBag", "bB" }, "a bag", "This is a blue bag....")

        }

        [Test]
        public void LocatesNothingTest()
        {
            Assert.IsNull(_bag.Locate("cap"));

        }


        [Test]
        public void LocatesItemTest()
        {
            _bag.Inventory.Put(_mirror);

            Assert.AreEqual(_mirror, _bag.Locate("mirror"));

        }


        [Test]
        public void FullDescTest()
        {
            _bag.Inventory.Put(_mirror);
            Assert.AreEqual("\tIn a bag you can see : \ta mirror (mirror)\n\n", _bag.FullDescrip

            //_player.FullDescription, "You're carrying: " + _inventory.ItemList);

        }

        [Test]
        public void LocatesItselfTest()
        {
            Assert.AreEqual(_bag, _bag.Locate("yellowBag"));

        }

```