Swinburne University of Technology

Object Oriented Programming (2022 S1)

Doubtfire Submission

Task 4.2P: Case Study Iteration 2: Player Class and Inventory

Submitted By:
Vaissheenavi Prabakaran
103508183
2022/04/23 15:49

Tutor:
Jai Cornes

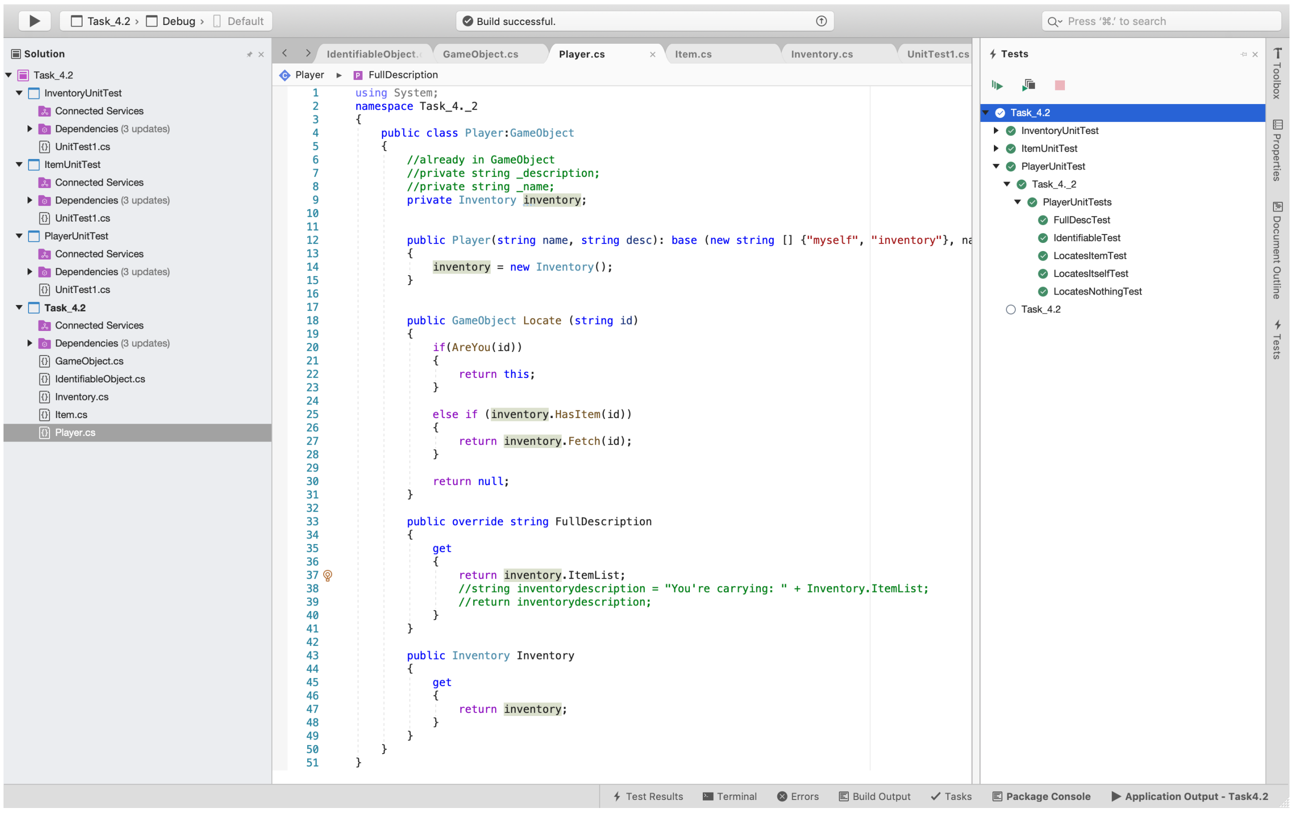
April 23, 2022

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


namespace Task_4._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 "";
            }
        }

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

            return;
        }



    }

}
```

```
using System;

namespace Task_4._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_4._2
{
    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 NUnit.Framework;

namespace Task_4._2
{
    public class PlayerUnitTests
    {
        public Player _player;
        public Item _mirror;
        public Inventory _inventory;


        [SetUp]
        public void Setup()
        {
            _inventory = new Inventory();
            _player = new Player("Fred", "You're carrying: ");
            _mirror = new Item(new string[] { "mirror", "comb" }, "a mirror", "This
             ↪  is a room item....");
        }


        [Test]
        public void IdentifiableTest()
        {
            Assert.IsTrue(_player.AreYou("myself"));
            Assert.IsTrue(_player.AreYou("inventory"));
        }


        [Test]
        public void LocatesNothingTest()
        {
            Assert.IsNull(_player.Locate("handbag"));

        }


        [Test]
        public void LocatesItemTest()
        {
            _inventory.Put(_mirror);
            _player.Locate("mirror");
            Assert.AreEqual("\ta mirror (mirror)\n", _inventory.ItemList);
            Assert.IsTrue(_inventory.HasItem("mirror"));
        }


        [Test]
        public void FullDescTest()
        {
            _inventory.Put(_mirror);
            StringAssert.Contains("You're carrying: " + _player.FullDescription,
             ↪  "You're carrying: " + _inventory.ItemList);

        }

        [Test]
        public void LocatesItselfTest()
        {
            Assert.AreEqual(_player, _player.Locate("myself"));
            Assert.AreEqual(_player, _player.Locate("inventory"));
        }
    }

}
```



Build successful.

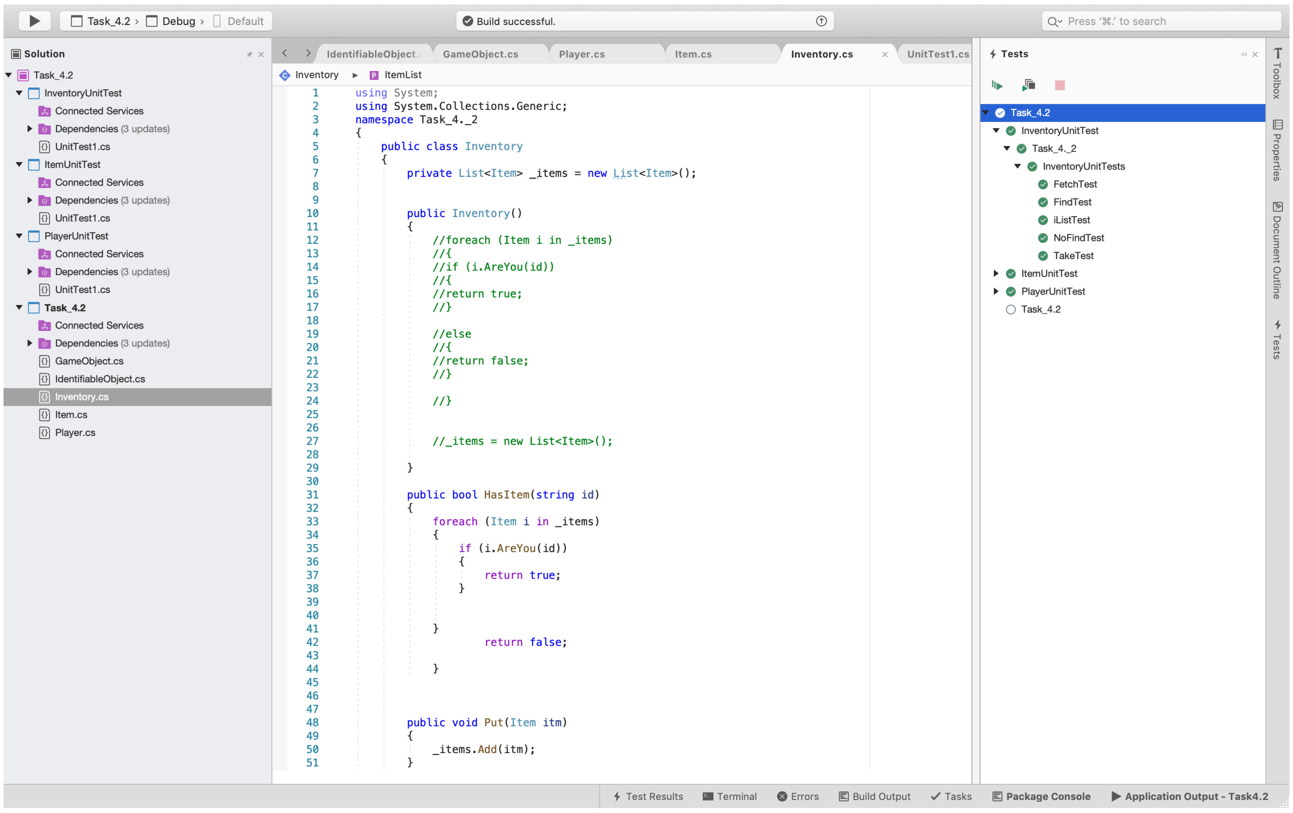
```
using System;
namespace Task_4._2
{
    public class Player:GameObject
    {
        //already in GameObject
        //private string _description;
        //private string _name;
        private Inventory inventory;


        public Player(string name, string desc): base (new string [] {"myself", "inventory"}, na
        {
            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;
            }
        }
    }
}
```

Tests
- ✓ Task_4.2
  - ✓ InventoryUnitTest
  - ✓ ItemUnitTest
  - ✓ PlayerUnitTest
    - ✓ Task_4._2
      - ✓ PlayerUnitTests
        - ✓ FullDescTest
        - ✓ IdentifiableTest
        - ✓ LocatesItemTest
        - ✓ LocatesItselfTest
        - ✓ LocatesNothingTest
  - ○ Task_4.2

```
using System;
using System.Collections.Generic;
namespace Task_4._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);
        }


        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;
            }

        }
    }
```

```
}
```

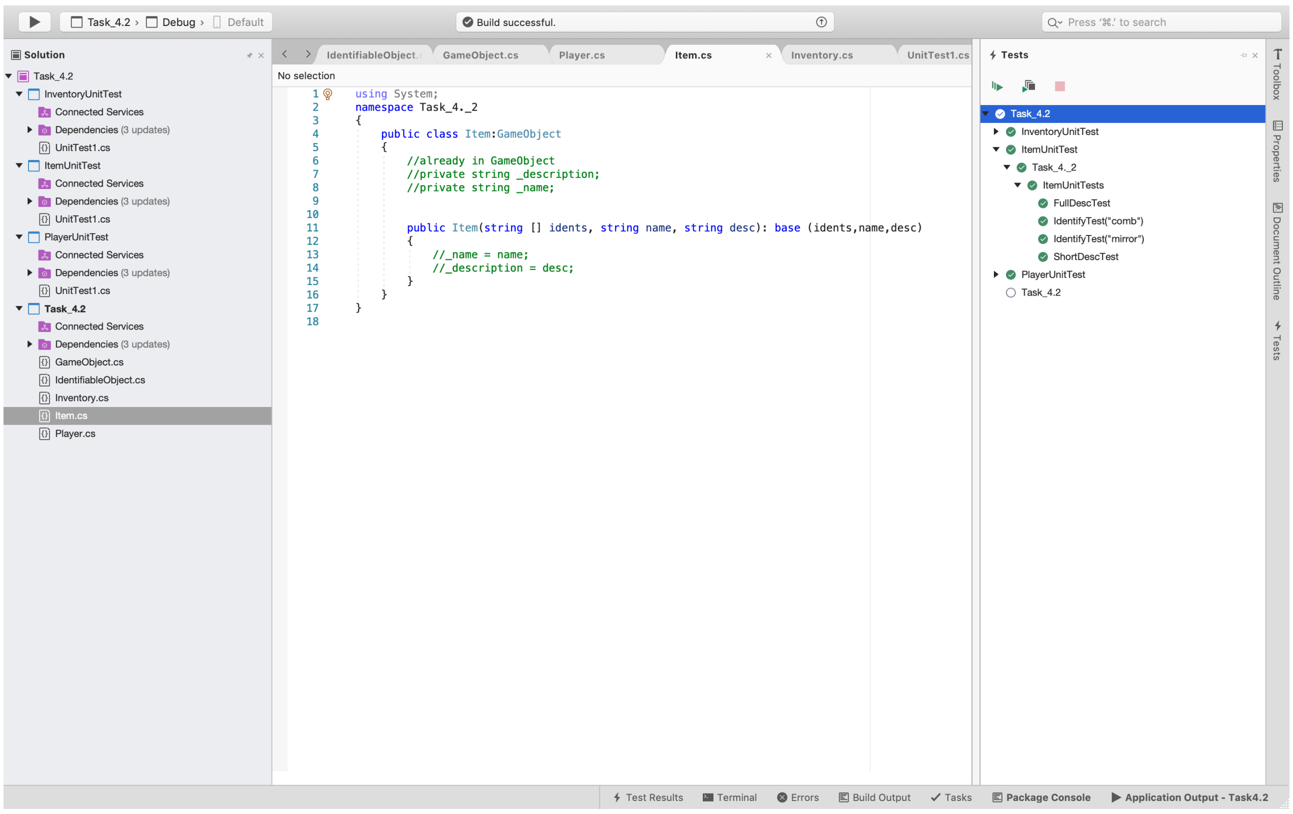
```
using NUnit.Framework;

namespace Task_4._2
{
    public class InventoryUnitTests
    {
        public Inventory _item;
        public Item comb;

        [SetUp]
        public void Setup()
        {
            _item = new Inventory();
            //_item.Put(comb);
            comb = new Item(new string[] { "comb" }, "a green comb", "This is a
             ↪  hair item....");
            _item.Put(comb);
        }


        [Test]
        public void NoFindTest()
        {
            Assert.IsFalse(_item.HasItem("handbag"));
        }


        [Test]
        public void iListTest()
        {
            //_item.Put(comb);
            //Assert.AreEqual("\t" + "a green comb (comb)" + "\n", _item.ItemList);

            string ItemLists = _item.ItemList;
            Assert.AreEqual("\t" + "a green comb (comb)" + "\n", ItemLists);
        }


        [Test]
        public void TakeTest()
        {
            _item.Take("comb");
            Item Comb = _item.Fetch("mirror");
            Assert.IsNull(Comb);

        }


        [Test]
        public void FindTest()
        {
            _item.Put(comb);
            Assert.IsTrue(_item.HasItem("comb"));
        }


        [Test]
        public void FetchTest()
        {
           _item.Put(comb);
            Assert.AreEqual(comb, _item.Fetch("comb"));
            Assert.IsTrue(_item.HasItem("comb"));
        }
    }
}
```



```
using System;
using System.Collections.Generic;
namespace Task_4._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);
        }
```

```
using System;
namespace Task_4._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 NUnit.Framework;

namespace Task_4._2
{
    public class ItemUnitTests
    {
        public Item _items;


        [SetUp]
        public void Setup()
        {
            _items = new Item(new string[] { "mirror", "comb" }, "a mirror", "This
             ↪  might be fine....");
        }

        [Test]
        public void ShortDescTest()
        {
            Assert.AreEqual("a mirror (mirror)", _items.ShortDescription);
        }


        [Test]
        public void FullDescTest()
        {
            Assert.AreEqual(_items.FullDescription, "This might be fine....");
        }


        [Test]
        [TestCase("mirror")]
        [TestCase("comb")]

        public void IdentifyTest(string id)
        {
            Assert.IsTrue(_items.AreYou(id));
        }
    }
}
```



```
Task_4.2
    InventoryUnitTest
        Connected Services
        Dependencies (3 updates)
        UnitTest1.cs
    ItemUnitTest
        Connected Services
        Dependencies (3 updates)
        UnitTest1.cs
    PlayerUnitTest
        Connected Services
        Dependencies (3 updates)
        UnitTest1.cs
    Task_4.2
        Connected Services
        Dependencies (3 updates)
        GameObject.cs
        IdentifiableObject.cs
        Inventory.cs
        Item.cs
        Player.cs
```

```
using System;
namespace Task_4._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;
        }
    }
}
```

```
✓ Task_4.2
    ✓ InventoryUnitTest
    ✓ ItemUnitTest
        ✓ Task_4._2
            ✓ ItemUnitTests
                ✓ FullDescTest
                ✓ IdentifyTest("comb")
                ✓ IdentifyTest("mirror")
                ✓ ShortDescTest
    ✓ PlayerUnitTest
    ○ Task_4.2
```