Pass Task 17 – Case Study 3

Related Learning Outcomes

ULO1 – Explain the OO Principles

This exercise demonstrated object encapsulation teaching me about how you can make features either public or private to only exist within the object. The topic taught me to convert abstract plans such as UML diagrams into actual code to use. The exercise used a lot of polymorphism to share similar methods and fields that classes shared from the Item object to the Bag.

ULO2 – Use OO Language and Library

Demonstrated class and constructor declaration, the use of conditional statements (e.g. “if”), and assigning values to parameters. The task examines how fields can be used by an object to remember information. We used a Property to get the FullDescription and Inventory from the counter and be able to access it outside the object.

ULO3 – Design, Develop and Test using an IDE

The code was developed using Xamarin Studio to build and run the program, as well as integrated debugging features to step and inspect values. Nunit was used for the unit testing for the Bag class.

ULO4 – Communicate using UML Diagrams

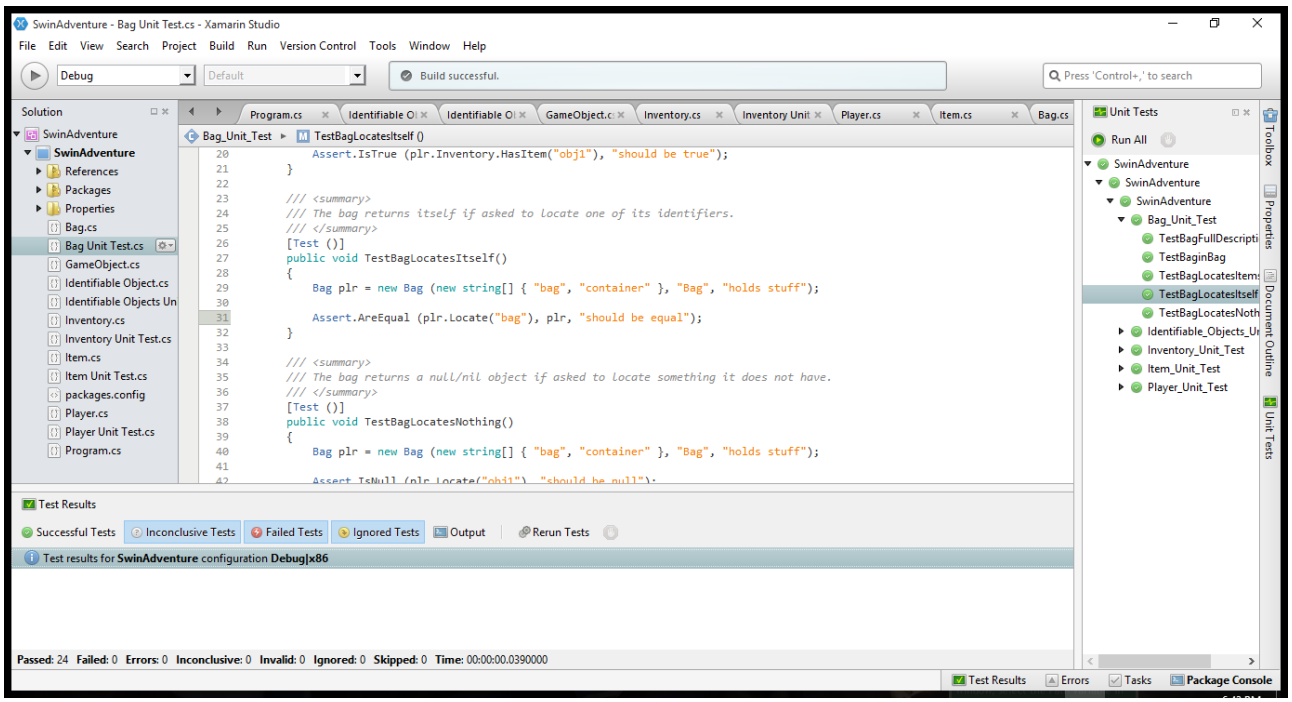
I learned how to interpret a UML class diagram and write the related code.

ULO5 – Describe Elements of Good OO Design

The exercise demonstrated correct use of C# coding conventions including proper naming, syntax and correct layout.

Screenshots

[use of IDE]


SwinAdventure - Bag Unit Test.cs - Xamarin Studio
File Edit View Search Project Build Run Version Control Tools Window Help
Debug
Default
Build successful.
Press 'Control+,' to search
Solution
SwinAdventure
SwinAdventure
References
Packages
Properties
Bag.cs
Bag Unit Test.cs
GameObject.cs
Identifiable Object.cs
Identifiable Objects Un
Inventory.cs
Inventory Unit Test.cs
Item.cs
Item Unit Test.cs
packages.config
Player.cs
Player Unit Test.cs
Program.cs
Program.cs
Identifiable O
Identifiable O
GameObject.c
Inventory.cs
Inventory Unit
Player.cs
Item.cs
Bag.cs
Bag_Unit_Test
TestBagLocatesItself ()
20 Assert.IsTrue (plr.Inventory.HasItem("obj1"), "should be true");
21 }
22
23 /// <summary>
24 /// The bag returns itself if asked to locate one of its identifiers.
25 /// </summary>
26 [Test ()]
27 public void TestBagLocatesItself()
28 {
29 Bag plr = new Bag (new string[] { "bag", "container" }, "Bag", "holds stuff");
30
31 Assert.AreEqual (plr.Locate("bag"), plr, "should be equal");
32 }
33
34 /// <summary>
35 /// The bag returns a null/nil object if asked to locate something it does not have.
36 /// </summary>
37 [Test ()]
38 public void TestBagLocatesNothing()
39 {
40 Bag plr = new Bag (new string[] { "bag", "container" }, "Bag", "holds stuff");
41
Test Results
Successful Tests
Inconclusive Tests
Failed Tests
Ignored Tests
Output
Rerun Tests
Test results for SwinAdventure configuration Debug|x86
Passed: 24 Failed: 0 Errors: 0 Inconclusive: 0 Invalid: 0 Ignored: 0 Skipped: 0 Time: 00:00:00.0390000
Unit Tests
Run All
SwinAdventure
SwinAdventure
SwinAdventure
Bag_Unit_Test
TestBagFullDescripti
TestBaginBag
TestBagLocatesItem
TestBagLocatesItself
TestBagLocatesNoth
Identifiable_Objects_U
Inventory_Unit_Test
Item_Unit_Test
Player_Unit_Test
Toolbox
Properties
Document Outline
Unit Tests
Test Results
Errors
Tasks
Package Console