Swinburne University of Technology

COS20007 Object Oriented Programming

Case Study - Iteration 8 - Command Processor

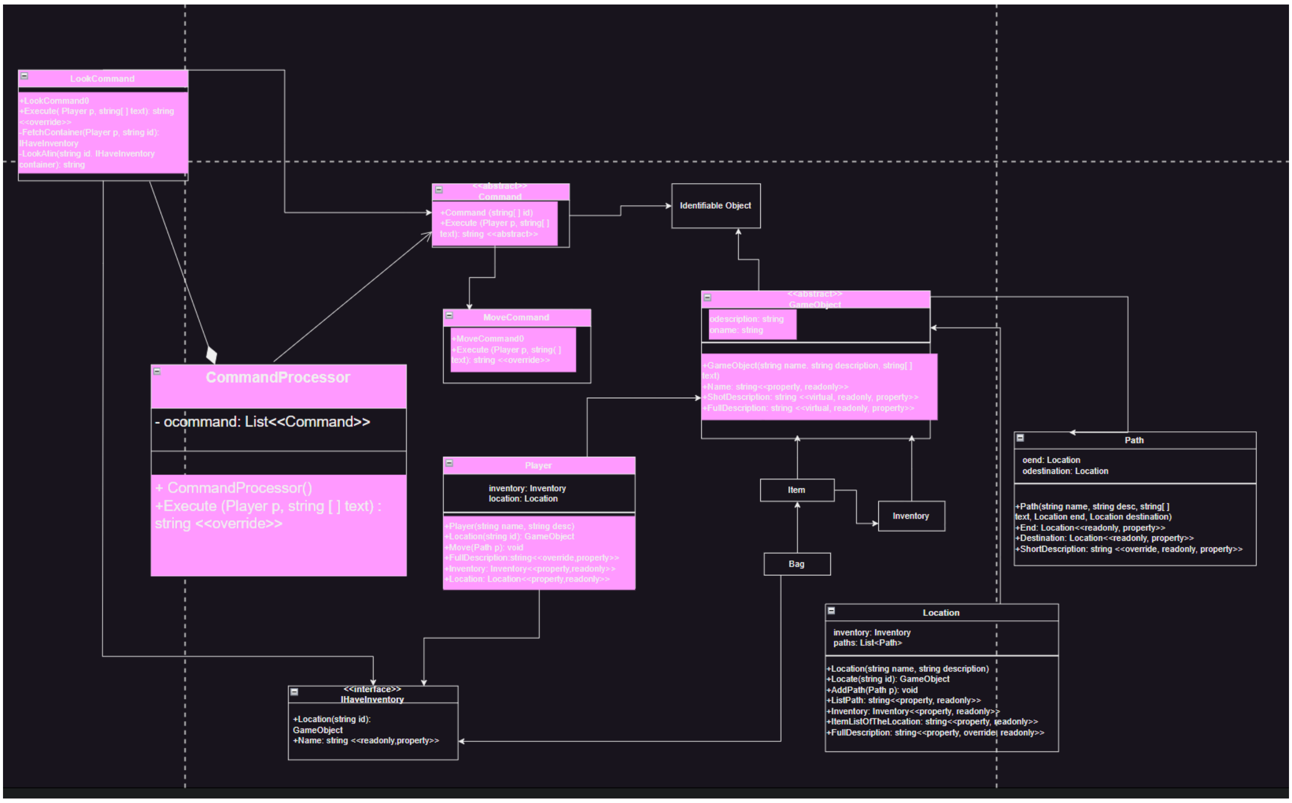
PDF generated at 18:45 on Sunday 26th November, 2023

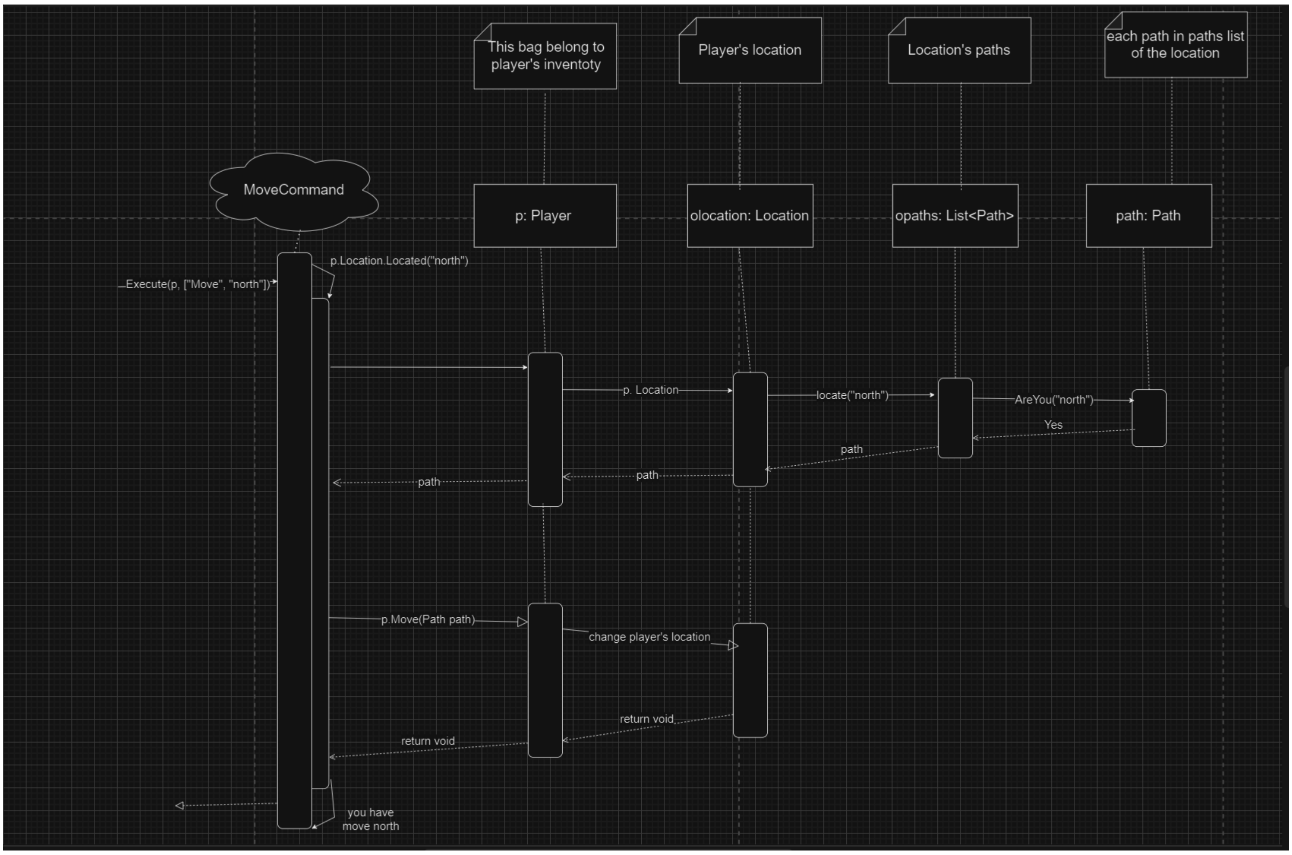
Produced by OOP Submission Formatter

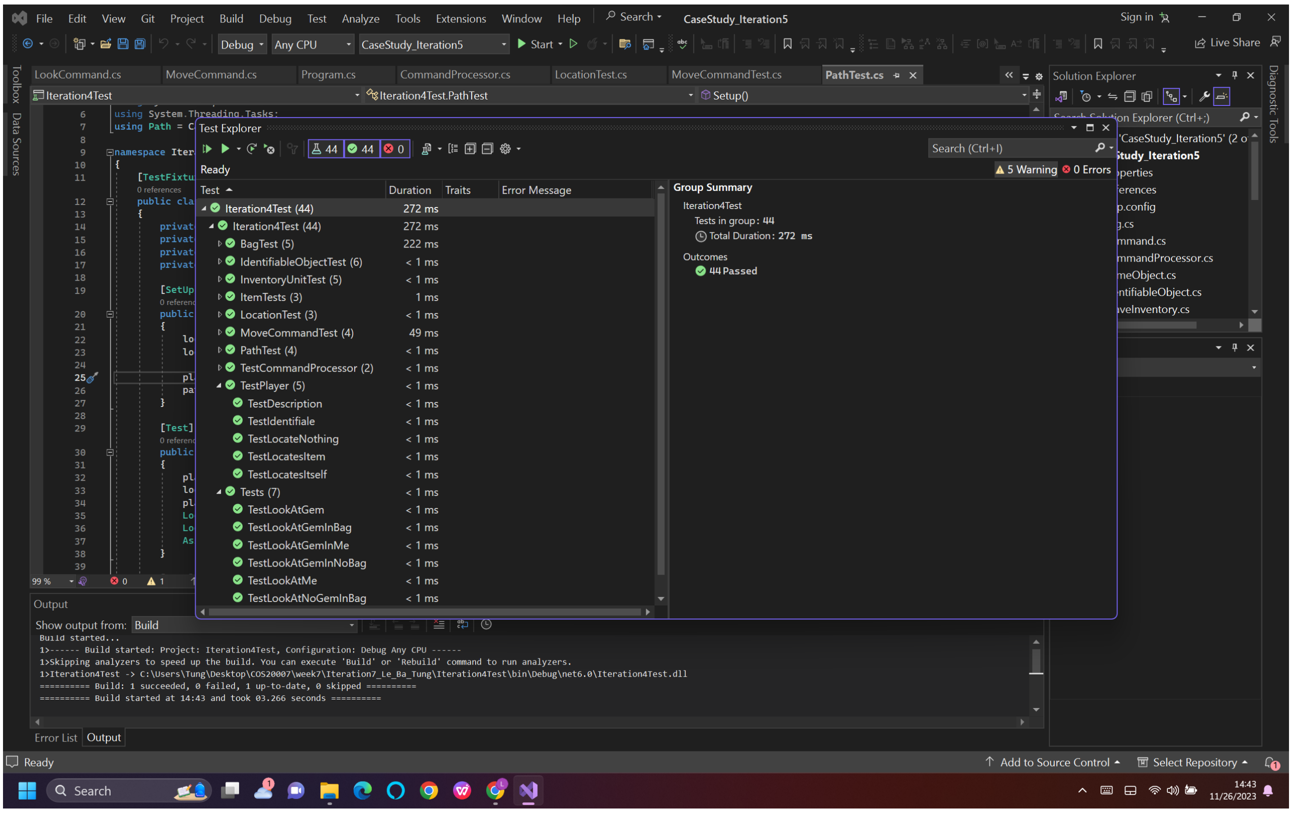
```csharp
using CaseStudy_Iteration5;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Linq;
using System.Text;
using System.Text.RegularExpressions;
using System.Threading.Tasks;

namespace Casestudy_Iteration5
{
    public class Program
    {
        public static void Main(string[] args)
        {
            Console.WriteLine("Welcome to Zombie War III");
            Console.WriteLine("Enter your name: ");
            string name = Console.ReadLine();
            Console.WriteLine("Hello " + name + "." + " Welcome to our game");
            Console.WriteLine("Let's me take some description about your:");
            string description = Console.ReadLine();
            Player player = new Player(name, description);

            Location location = new Location("military training grounds", "Military area");
            player.Location = location;
            Location loc = new Location("military base", "this is a large military area and you just see a gun in the table ");
            Path path = new Path(new string[] { "downstair" }, "stair ", "go to the stair", location, loc);
            Path path2 = new Path(new string[] { "west" }, "f22", "go to f22 to find the military airbase", loc, location);
            Location loc2 = new Location("airbase", "this is a lot of war plane in here");
            Path path3 = new Path(new string[] { "north" }, "b52", "go to the b52", location, loc2);
            Path path4 = new Path(new string[] { "south" }, "door", "go to the door to move the next area", loc2, location);
            Location loc3 = new Location("billet", "this is a place solidiers live");
            Path path5 = new Path(new string[] { "upstair" }, "stair", "go upstair to find the map", location, loc3);
            Path path6 = new Path(new string[] { "west" }, "red house", "go to red house to find the billet", loc3, location);
            location.AddPath(path);
            loc.AddPath(path2);
            location.AddPath(path3);
            loc2.AddPath(path4);
            location.AddPath(path5);
            loc3.AddPath(path6);
            Item item1 = new Item(new string[] { "b52" }, "warplane", "this is the biggest plane");
            player.Inventory.Put(item1);
            Item item2 = new Item(new string[] { "m4a1" }, "auto", "this is a powerful gun");
            player.Inventory.Put(item2);
            Item item3 = new Item(new string[] { "gold" }, "knife", "this is a sharp knife");
            player.Inventory.Put(item3);
            Item item4 = new Item(new string[] { "compass" }, "mititary map", "this is a useful thing");
            player.Inventory.Put(item4);
            Bag bag = new Bag(new string[] { "huron" }, "Military airbase", "this is a big plane");

            player.Inventory.Put(bag);
            bag.Inventory.Put(item1);
            location.Inventory.Put(item1);
            loc2.Inventory.Put(item2);
            loc.Inventory.Put(item3);
            loc3.Inventory.Put(item4);

            string text;
            //    do
            //    {
            //        Console.WriteLine("Put your command here:  ");
            //        text = Console.ReadLine();

            //        Console.WriteLine(lk.Execute(player, text));
            //    } while (text != "out");
            //}
            var lk = new CommandProcessor();

            do
            {
                Console.WriteLine("Put your command here: ");
                text = Console.ReadLine();
                Console.WriteLine(lk.Execute(text, player));
            } while (text != "out");

        }
    }
}
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace CaseStudy_Iteration5
{
    public class CommandProcessor
    {
        private List<Command> ocmd;
        public CommandProcessor()
        {
            ocmd = new List<Command>();
            ocmd.Add(new MoveCommand());
            ocmd.Add(new LookCommand());
        }
        public string Execute(string cmd, Player player)
        {

            string[] array = cmd.Split(' ');

            if (cmd.Length == 0)
            {
                return "I can see this command ";
            }

            if (array[0] == "out")
            {
                return "bai bai. See you again";
            }
            Command matchedCommand = null;

            foreach (Command command in ocmd)
            {
                if (command.AreYou(array[0].ToLower()))
                {
                    matchedCommand = command;
                    break;
                }
            }

            if (matchedCommand == null)
            {
                return "I dont know what do you want. Please enter another command";
            }

            return matchedCommand.Execute(player, array);
        }
    }

}
```

```
using CaseStudy_Iteration5;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using Path = CaseStudy_Iteration5.Path;
using System.Threading.Tasks;

namespace Iteration4Test
{
    [TestFixture]
    public class TestCommandProcessor
    {
        private CommandProcessor ocp;
        private Player oplayer;
        private Location olocation, olocation1;
        private Item item;
        private Path path;

        [SetUp]
        public void SetUp()
        {
            ocp = new CommandProcessor();
            oplayer = new Player("Tung", "best player");
            item = new Item(new string[] { "su30" }, "warplane", "this is power plane");
            oplayer.Inventory.Put(item);

            olocation1 = new Location("military training grounds", "Military area");
            olocation = new Location("airbase", "this is a lot of war plane in here");
            path = new Path(new string[] { "north" }, "b52", "go to the b52", olocation1, olocation);

            olocation.AddPath(path);
            oplayer.Location = olocation;

        }

        [Test]
        public void TestLookCommand()
        {
            Assert.That(ocp.Execute("look at su30", oplayer), Is.EqualTo("this is power plane"));
        }

        [Test]
        public void TestMoveCommand()
        {
            string result = ocp.Execute("move north", oplayer);
            Assert.AreEqual("If you want to go to north " + "going to " + path.Name + " and reached the " + oplayer.Location.Name + "\n\n" + oplayer.Location.FullDescription, result);
            Assert.AreEqual(olocation, oplayer.Location);
            Assert.That(oplayer.Location, Is.EqualTo(olocation));
        }




    }
}
```


LookCommand
+LookCommand0
+Execute(Player p, string[ ] text): string <<override>>
-FetchContainer(Player p, string id): IHaveInventory
-LookAtin(string id. IHaveInventory container): string
<<abstract>>
Command
+Command (string[ ] id)
+Execute (Player p, string[ ] text): string <<abstract>>
Identifiable Object
MoveCommand
+MoveCommand0
+Execute (Player p, string( ] text): string <<override>>
<<abstract>>
GameObject
odescription: string
oname: string
+GameObject(string name. string description, string[ ] text)
+Name: string<<property, readonly>>
+ShotDescription: string <<virtual, readonly, property>>
+FullDescription: string <<virtual, readonly, property>>
CommandProcessor
- ocommand: List<<Command>>
+ CommandProcessor()
+Execute (Player p, string [ ] text) : string <<override>>
Player
inventory: Inventory
location: Location
+Player(string name, string desc)
+Location(string id): GameObject
+Move(Path p): void
+FullDescription:string<<override,property>>
+Inventory: Inventory<<property,readonly>>
+Location: Location<<property,readonly>>
Item
Inventory
Bag
Path
oend: Location
odestination: Location
+Path(string name, string desc, string[ ] text, Location end, Location destination)
+End: Location<<readonly, property>>
+Destination: Location<<readonly, property>>
+ShortDescription: string <<override, readonly, property>>
Location
inventory: Inventory
paths: List<Path>
+Location(string name, string description)
+Locate(string id): GameObject
+AddPath(Path p): void
+ListPath: string<<property, readonly>>
+Inventory: Inventory<<property, readonly>>
+ItemListOfTheLocation: string<<property, readonly>>
+FullDescription: string<<property, override, readonly>>
<<interface>>
IHaveInventory
+Location(string id): GameObject
+Name: string <<readonly,property>>


This bag belong to player's inventoty
Player's location
Location's paths
each path in paths list of the location
MoveCommand
p: Player
olocation: Location
opaths: List<Path>
path: Path
p.Location.Located("north")
Execute(p, ["Move", "north"])
p. Location
locate("north")
AreYou("north")
Yes
path
path
path
p.Move(Path path)
change player's location
return void
return void
you have move north

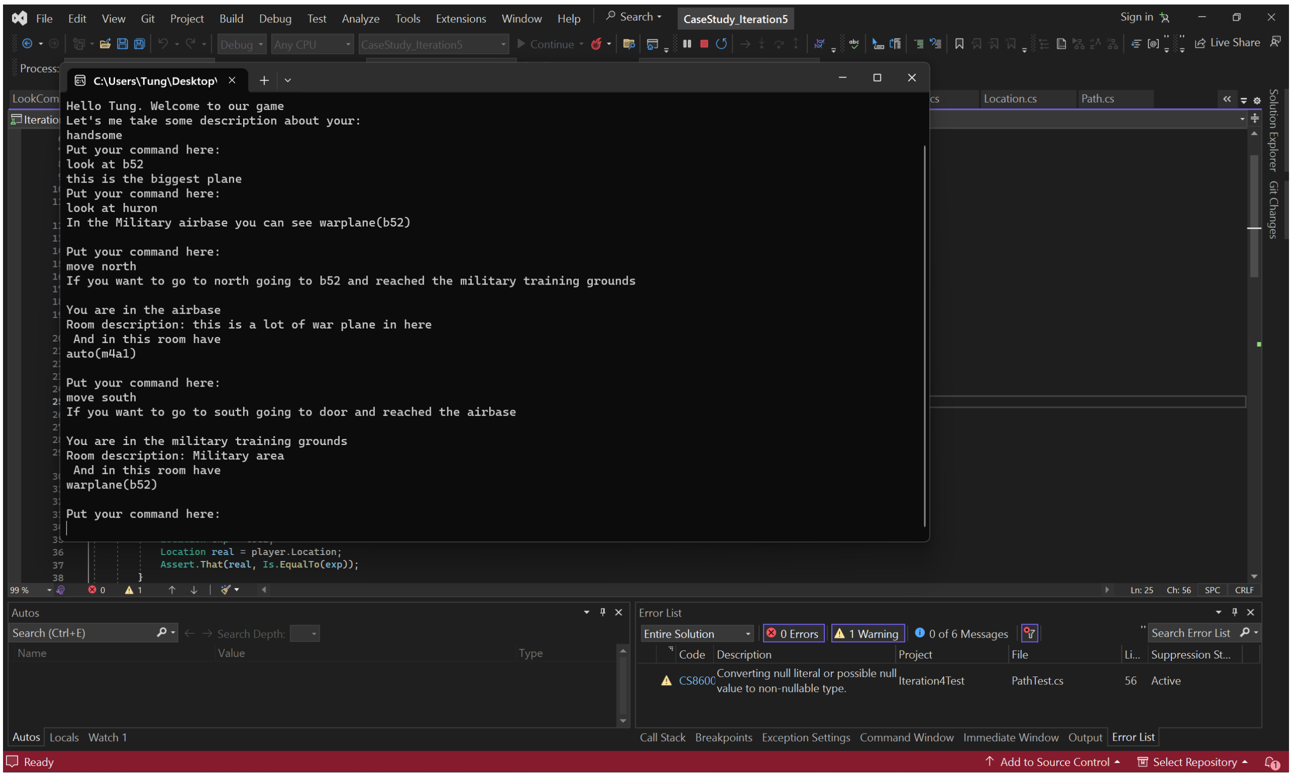


```
Test Explorer
44   44   0
Ready                                          5 Warning   0 Errors

Test                         Duration   Traits   Error Message
Iteration4Test (44)           272 ms
  Iteration4Test (44)         272 ms
    BagTest (5)               222 ms
    IdentifiableObjectTest (6)  < 1 ms
    InventoryUnitTest (5)     < 1 ms
    ItemTests (3)               1 ms
    LocationTest (3)          < 1 ms
    MoveCommandTest (4)        49 ms
    PathTest (4)              < 1 ms
    TestCommandProcessor (2)  < 1 ms
    TestPlayer (5)            < 1 ms
      TestDescription         < 1 ms
      TestIdentifiale         < 1 ms
      TestLocateNothing       < 1 ms
      TestLocatesItem         < 1 ms
      TestLocatesItself       < 1 ms
    Tests (7)                 < 1 ms
      TestLookAtGem           < 1 ms
      TestLookAtGemInBag      < 1 ms
      TestLookAtGemInMe       < 1 ms
      TestLookAtGemInNoBag    < 1 ms
      TestLookAtMe            < 1 ms
      TestLookAtNoGemInBag    < 1 ms

Group Summary
Iteration4Test
  Tests in group: 44
  Total Duration: 272 ms
Outcomes
  44 Passed
```

```
Output
Show output from: Build
Build started...
1>------ Build started: Project: Iteration4Test, Configuration: Debug Any CPU ------
1>Skipping analyzers to speed up the build. You can execute 'Build' or 'Rebuild' command to run analyzers.
1>Iteration4Test -> C:\Users\Tung\Desktop\COS20007\week7\Iteration7_Le_Ba_Tung\Iteration4Test\bin\Debug\net6.0\Iteration4Test.dll
========== Build: 1 succeeded, 0 failed, 1 up-to-date, 0 skipped ==========
========== Build started at 14:43 and took 03.266 seconds ==========
```

Ready

14:43
11/26/2023



```
Hello Tung. Welcome to our game
Let's me take some description about your:
handsome
Put your command here:
look at b52
this is the biggest plane
Put your command here:
look at huron
In the Military airbase you can see warplane(b52)

Put your command here:
move north
If you want to go to north going to b52 and reached the military training grounds

You are in the airbase
Room description: this is a lot of war plane in here
 And in this room have
auto(m4a1)

Put your command here:
move south
If you want to go to south going to door and reached the airbase

You are in the military training grounds
Room description: Military area
 And in this room have
warplane(b52)

Put your command here:
```

```
            Location real = player.Location;
            Assert.That(real, Is.EqualTo(exp));
        }
```

Entire Solution | 0 Errors | 1 Warning | 0 of 6 Messages | Search Error List

	Code	Description	Project	File	Li...	Suppression St...
⚠	CS8600	Converting null literal or possible null value to non-nullable type.	Iteration4Test	PathTest.cs	56	Active