Swinburne University of Technology

COS20007 Object Oriented Programming

Case Study - Iteration 7 - Paths

PDF generated at 11:36 on Monday 13th November, 2023

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

namespace Ass24
{
    public class Path: GameObject
    {
        private string _direction;

        Location _loc2;
        public Path(string direction, Location _loc1, Location loc2) :base(new string[] {direction},"Path","A Path")
        {
            _direction = direction;
            _loc2=loc2;


        }

        public string Direction
        {
            get { return _direction; }
        }

        public Location Destination
        {
            get { return _loc2; }
        }
    }

}
```

```
using System.Numerics;
using Ass24;

namespace MoveCommandTest
{
    public class Tests
    {
        Location _location1;
        Location _location2;
        Player player;
        Ass24.Path _path;
        MoveCommand _move;
        [SetUp]
        public void Setup()
        {
            player = new Player("lily", "Tired");
            _location1 = new Location(new string[] { "swin", "burne" }, "Swinburne", "A place to study.");
            _location2 = new Location(new string[] { "home", "house" }, "Home", "A place to sleep.");
            _path = new Ass24.Path("North", _location1, _location2);
            _location1.AddPath(_path);
            _move = new MoveCommand();
            player.Location = _location1;
        }

        [Test]
        public void PathName()
        {
            Assert.AreEqual(player.Location.FullDescription, _location1.FullDescription);
        }

        [Test]
        public void PathDirection()
        {
            Assert.AreEqual(_path.Direction, "North");
        }

        [Test]
        public void PathDestination()
        {
            Assert.AreEqual(_path.Destination, _location2);
        }

    }
}
```

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

namespace Ass24
{
    public class Location :GameObject, IHaveInventory
    {

        private string _name;
        private string _desc;
        Inventory _inventory;
        List<Path> _paths;

        public Location(string[] id, string name, string desc) :base(id, name, desc)
        {
            _inventory = new Inventory();
            _name = name;
            _desc = desc;
            _paths = new List<Path>();
        }


        public GameObject Locate(string itemid)
        {
            if (AreYou(itemid))
            {
                return this;
            }
            foreach (Path path in _paths)
            {
                if (path.Direction == itemid)
                { return path;}
            }
            return _inventory.Fetch(itemid);
        }


        public override string FullDescription
        {
            get { return $"You are at {_name}, {_desc}"; }
        }
        public Inventory Inventory
        {
            get { return _inventory; }

        }


        public void AddPath(Path path)
```

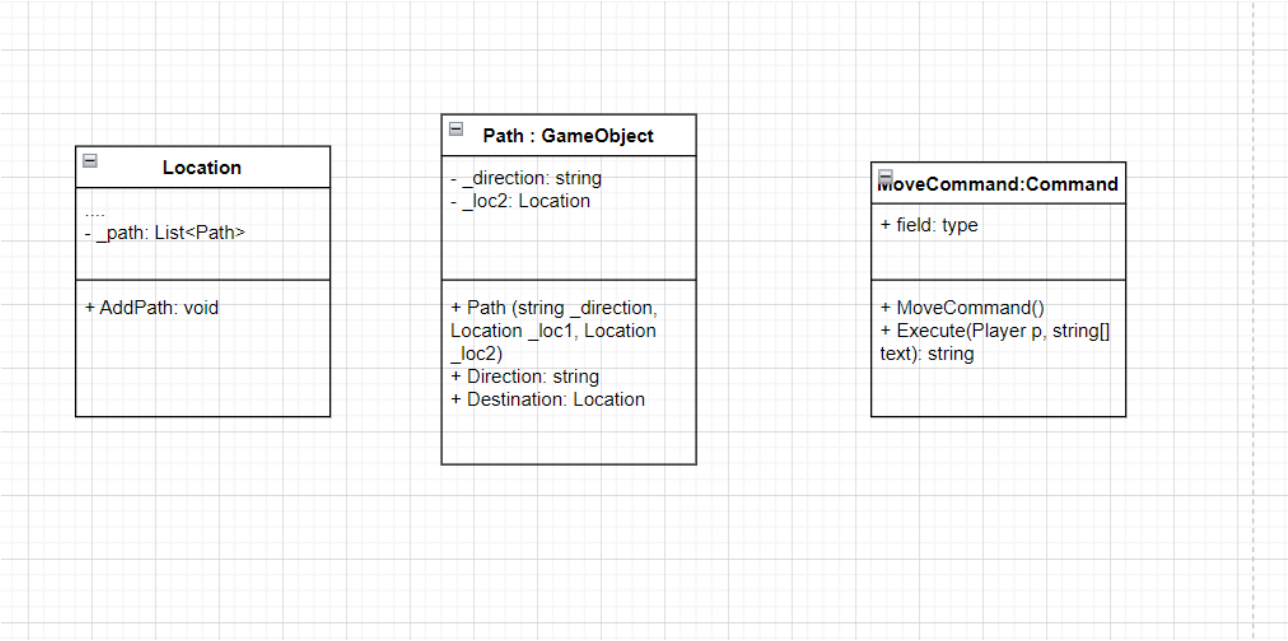
```
        {
            _paths.Add(path);
        }


    }
}
```

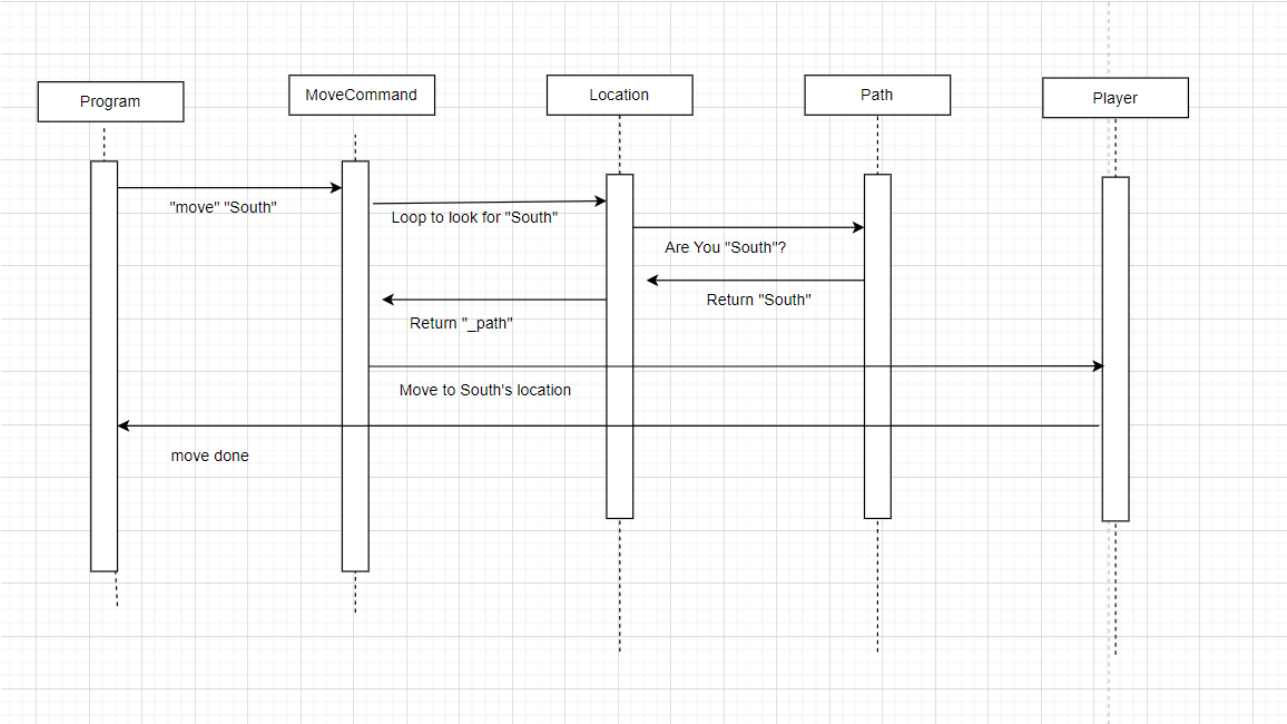
```
using Ass24;
using System.Numerics;

namespace LocationTest
{
    public class Tests
    {
        Location _location;
        Player player;
        Item spoon;
        Item plate;


        [SetUp]


        public void Setup()
        {
            player = new Player("lily", "Tired");

            spoon = new Item(new string[] { "spoon" }, "a spoon", "Can be used for eating but also as an inefficent weapon");
            plate = new Item(new string[] { "plate" }, "a plate", "Simple plate");
            _location = new Location(new string[] {"blue","lock" }, "Blue Lock", "A sport stadium");
            _location.Inventory.Put(spoon);
            _location.Inventory.Put(plate);
        }

        [Test]
        public void SelfIdentify()
        {
            Assert.AreEqual(_location.Locate(_location.FirstId),_location);
        }

        [Test]
        public void ItemIdentify()
        {
            Assert.AreEqual(_location.Locate(spoon.FirstId), spoon);
            Assert.AreEqual(_location.Locate(plate.FirstId), plate);
        }

        [Test]
        public void PlayerIdentify()
        {
            player.Location = _location;
            Assert.AreEqual(player.Locate(spoon.FirstId), spoon);
        }
    }
}
```

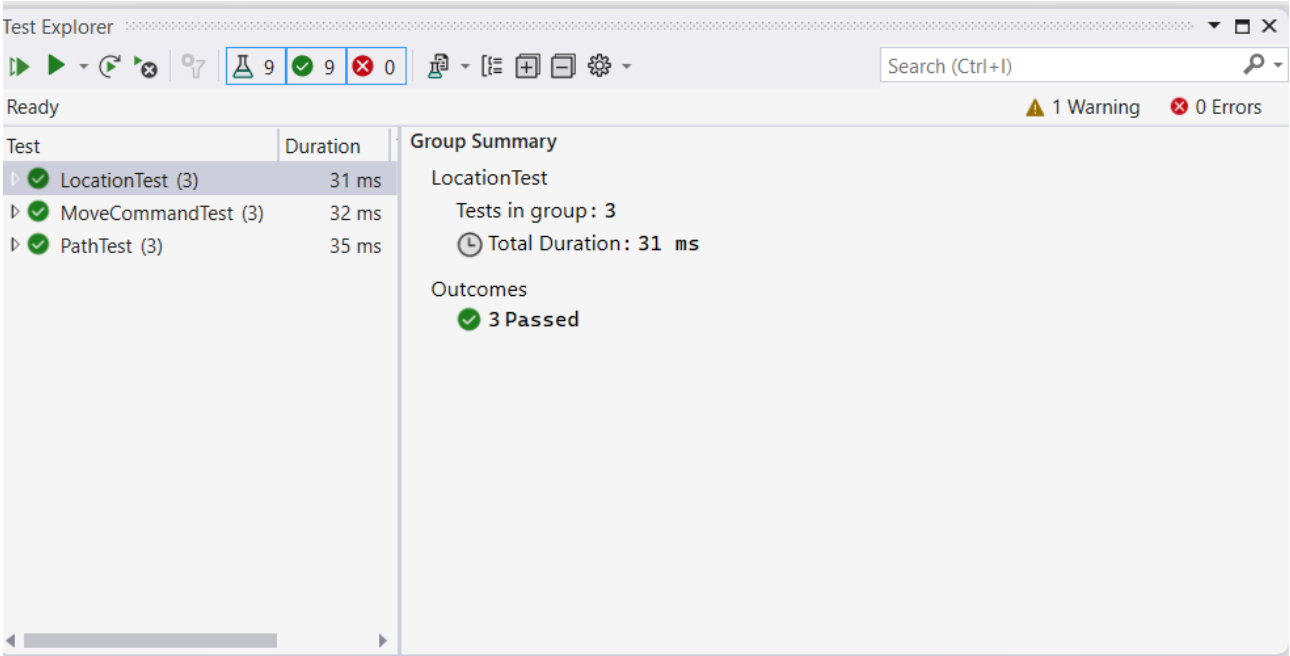
```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Ass24
{
    public class MoveCommand:Command
    {
        public MoveCommand() : base(new string[] { "move" })
        { }

        public override string Execute(Player p, string[] text)
        {
            string error = "Cannot execute command.";
            string _move;
            if (text[0].ToLower() == "move" || text[0].ToLower() == "go" || text[0].ToLower() == "leave" || text[0].ToLower() == "head")
            {
                switch(text.Length)
                {
                    case 1:
                        return "No destinantion";
                    case 2:
                        _move = text[1];
                        break;
                    case 3:
                        _move = text[2];
                        break;

                    default:
                        return error;
                }

                GameObject _path = p.Location.Locate(_move);
                if (_path != null)
                {
                    if (_path.GetType() != typeof(Path))
                    {
                        return "Could not find the " + _path.Name;
                    }
                    p.Move((Path)_path);
                    return "You have moved " + _move + " through a " + _path.Name + " to " + p.Location.Name;
                }
                return "Cannot execute because the path is null";




            }
```

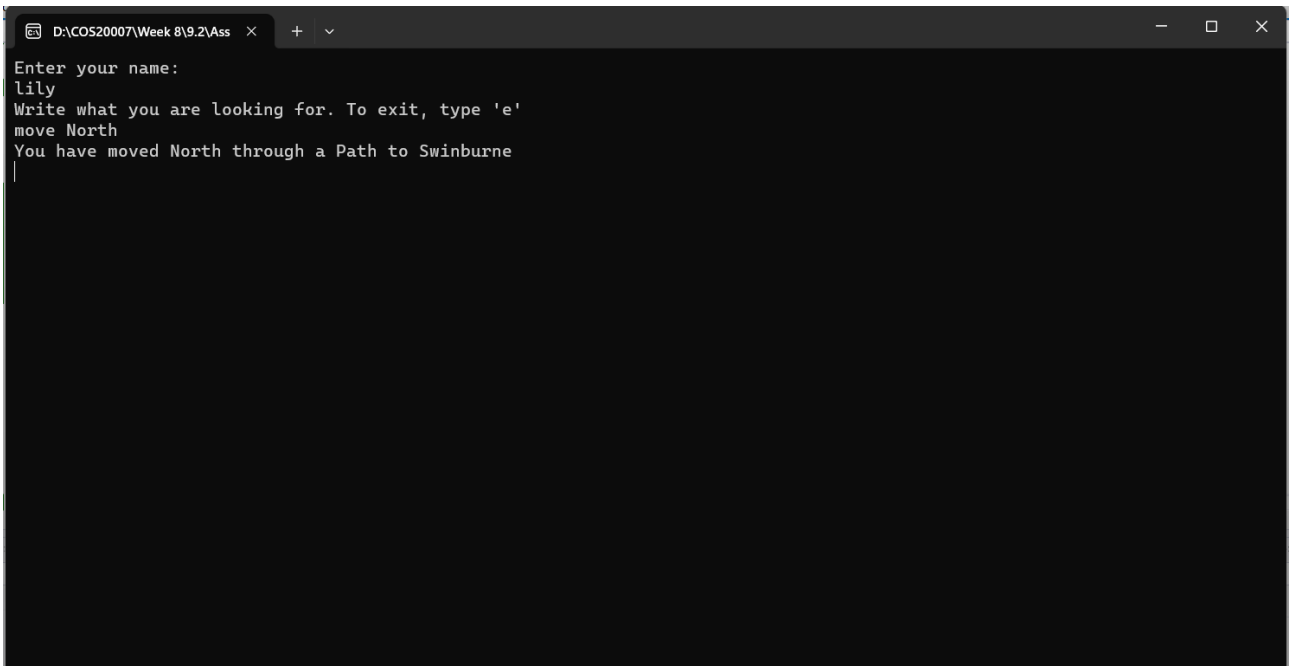
```
            else
                return "Invalid move command";
        }



    }
}
```

```
using Ass24;
using System.Numerics;

namespace PathTest
{
    public class Tests
    {
        Location _location1;
        Location _location2;
        Player player;
        Ass24.Path _path;
       MoveCommand _move;
        [SetUp]
        public void Setup()
        {
            player = new Player("lily", "Tired");
            _location1 = new Location(new string[] { "swin", "burne" }, "Swinburne", "A place to study.");
            _location2 = new Location(new string[] { "home", "house" }, "Home", "A place to sleep.");
            _path = new Ass24.Path("North", _location1,_location2);
            _location1.AddPath(_path);
            _move = new MoveCommand();
            player.Location = _location1;
        }

        [Test]
        public void PathName()
        {
            Assert.AreEqual(player.Location.FullDescription, _location1.FullDescription);
        }

        [Test]
        public void PathMove()
        {

            string result = _move.Execute(player, new string[] { "move", "North" });
            Assert.AreEqual($"You have moved {_path.Direction} through a {_path.Name} to {player.Location.Name}", result );
        }

        [Test]
        public void PathInvalid()
        {
            string result = _move.Execute(player, new string[] { "move", "South" });
            Assert.AreEqual("Cannot execute because the path is null", result);
        }
    }
}
```


Location
....
- _path: List<Path>
+ AddPath: void
Path : GameObject
- _direction: string
- _loc2: Location
+ Path (string _direction, Location _loc1, Location _loc2)
+ Direction: string
+ Destination: Location
MoveCommand:Command
+ field: type
+ MoveCommand()
+ Execute(Player p, string[] text): string


Program
MoveCommand
Location
Path
Player
"move" "South"
Loop to look for "South"
Are You "South"?
Return "South"
Return "_path"
Move to South's location
move done


Test Explorer
9
9
0
Search (Ctrl+I)
Ready
1 Warning
0 Errors
Test
Duration
LocationTest (3)
31 ms
MoveCommandTest (3)
32 ms
PathTest (3)
35 ms
Group Summary
LocationTest
Tests in group: 3
Total Duration: 31 ms
Outcomes
3 Passed


D:\COS20007\Week 8\9.2\Ass
Enter your name:
lily
Write what you are looking for. To exit, type 'e'
move North
You have moved North through a Path to Swinburne