

# Player.cs

...COS20007\Week\_7\7.2C\Iteration 6\Iteration1\Player.cs 1

---

```
1 namespace Iteration1
2 {
3     public class Player : GameObject, IHaveInventory
4     {
5         private Inventory _inventory;
6         private Location _location;
7
8         public Player(string name, string desc) : base(new string[] {"me", "inventory"}, name, desc)
9         {
10             _inventory = new Inventory();
11         }
12
13         public GameObject Locate(string id)
14         {
15             try
16             {
17                 if (AreYou(id))
18                 {
19                     return this;
20                 }
21                 else if (_inventory.HasItem(id))
22                 {
23                     return _inventory.Fetch(id);
24                 }
25                 else if (_location.Inventory.HasItem(id))
26                 {
27                     return _location.Locate(id);
28                 }
29             }
30             catch (Exception e)
31             {
32                 return null;
33             }
34             return null;
35         }
36
37         public override string FullDescription
38         {
39             get
40             {
41                 string fulldesc = "";
42                 fulldesc += $"You are {Name}, {base.FullDescription}\n";
43                 fulldesc += "You are carrying\n";
44                 fulldesc += $"{_inventory.ItemList}";
45                 return fulldesc;
46             }
47         }
48     }
```

```
49     public Location CurrentLocation
50     {
51         get => _location;
52         set => _location = value;
53     }
54
55     public Inventory Inventory
56     {
57         get => _inventory;
58     }
59 }
60 }
61 }
```

# Location.cs

...S20007\Week\_7\7.2C\Iteration 6\Iteration1\Location.cs

1

```
1 namespace Iteration1
2 {
3     public class Location : GameObject, IHaveInventory
4     {
5         private Inventory _inventory;
6
7         public Location(string[] ids, string name, string desc) : base(ids, name, desc)
8         {
9             _inventory = new Inventory();
10        }
11
12        public GameObject Locate(string id)
13        {
14            if (AreYou(id))
15            {
16                return this;
17            }
18            else if (_inventory.HasItem(id))
19            {
20                return _inventory.Fetch(id);
21            }
22            else
23            {
24                return null;
25            }
26        }
27
28        public Inventory Inventory
29        {
30            get => _inventory;
31        }
32    }
33}
34
```

# TestLocations.cs

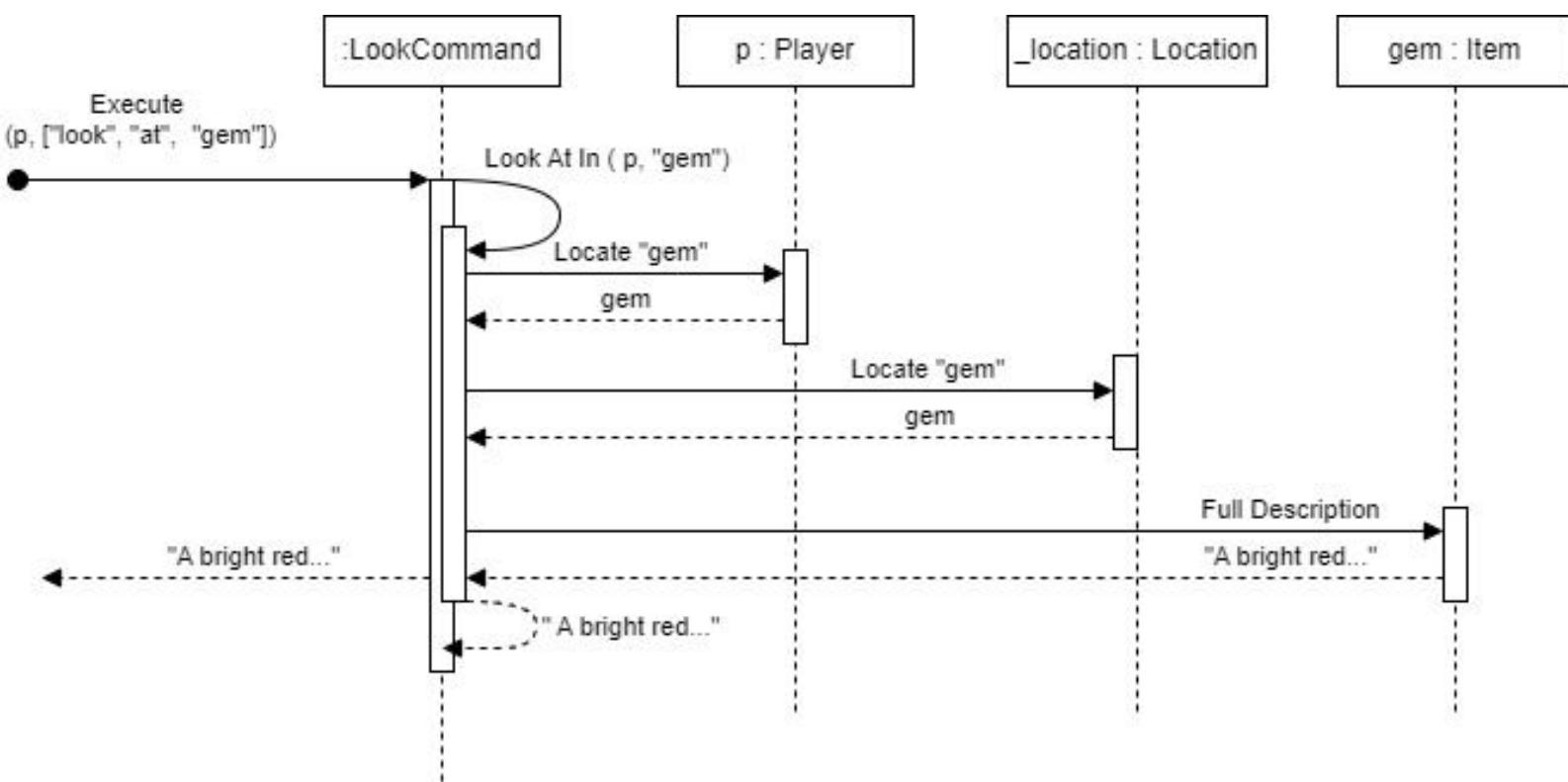
..07\Week\_7\7.2C\Iteration 6\TestLocations\UnitTest1.cs

1

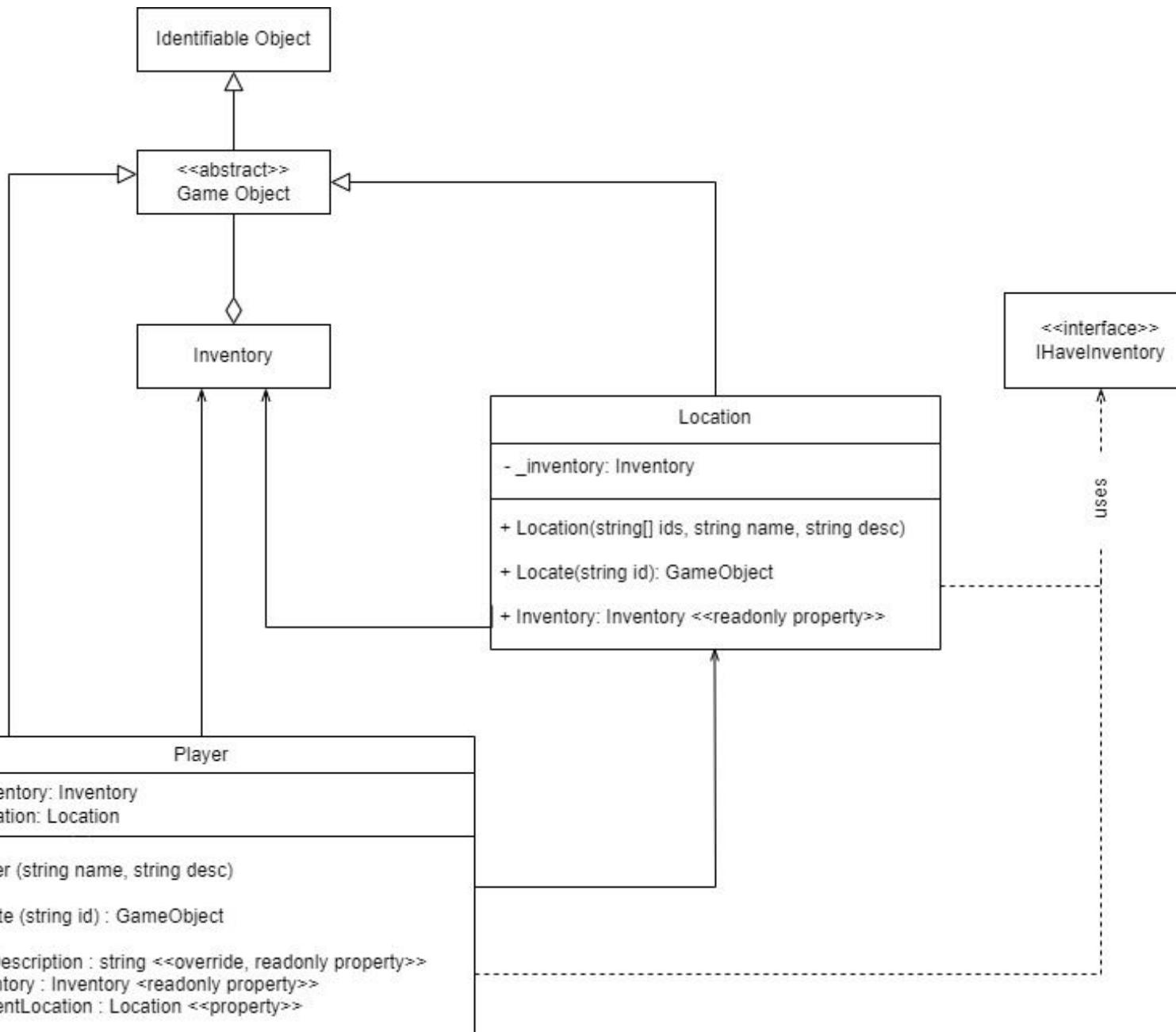
```
1  using Iteration1;
2
3  namespace TestLocations
4  {
5      public class Tests
6      {
7          Player _player;
8          Location _location;
9          Item _sword;
10         Item _ak47;
11         Item _grenade;
12
13         [SetUp]
14         public void Setup()
15         {
16             _player = new Player("Chien", "A boy with high curiosity");
17             _location = new Location(new string[] { "military base" },
18                 "military base", "large area");
19             _sword = new Item(new string[] { "sword", "melee" }, "sword",
20                 "Short range weapon");
21             _ak47 = new Item(new string[] { "ak47" }, "ak47", "Long range
22                 weapon");
23             _grenade = new Item(new string[] { "grenade" }, "grenade",
24                 "Very high damage weapon!");
25             _location.Inventory.Put(_sword);
26             _location.Inventory.Put(_ak47);
27             _player.CurrentLocation = _location;
28         }
29
30         [Test]
31         public void TestLocationLocatesItself()
32         {
33             Assert.That(_location.Locate("military base"), Is.EqualTo
34                 (_location));
35             Assert.Pass();
36         }
37
38         [Test]
39         public void TestLocationLocatesItem()
40         {
41             Assert.That(_location.Locate("sword"), Is.EqualTo(_sword));
42             Assert.Pass();
43         }
44
45         [Test]
46         public void TestLocationLocatesNothing()
47         {
48             Assert.That(_location.Locate("grenade"), Is.EqualTo(null));
49             Assert.Pass();
50         }
51     }
52 }
```

```
45    }
46
47    [Test]
48    public void TestPLayerLocatesItemInLocation()
49    {
50        Assert.That(_player.Locate("ak47"), Is.EqualTo(_ak47));
51        Assert.Pass();
52    }
53
54    [Test]
55    public void TestPlayerLocatesNothingInLocation()
56    {
57        Assert.That(_player.Locate("grenade"), Is.EqualTo(null));
58        Assert.Pass();
59    }
60}
61}
```

# Sequence Diagram



# UML Diagram



# Program.cs

```
..\OS20007\Week_7\7.2C\Iteration_6\Iteration1\Program.cs 1
1  using System.Runtime.InteropServices;
2
3  namespace Iteration1
4  {
5      internal class Program
6      {
7          static void Main(string[] args)
8          {
9              string name;
10             string desc_;
11             string command;
12             Location location;
13             Item redbull;
14             Item sword;
15             Item ak47;
16             Item grenade;
17
18             Console.WriteLine("Player name: ");
19             name = Console.ReadLine();
20             Console.WriteLine("Player description: ");
21             desc_ = Console.ReadLine();
22             Player player = new Player(name, desc_);
23
24             location = new Location(new string[] { "hospital" },
25                                     "hospital", "This is a state-of-the-art hospital");
26             redbull = new Item(new string[] { "Redbull" }, "Redbull",
27                               "Drink to be more energetic!");
28             location.Inventory.Put(redbull);
29             player.CurrentLocation = location;
30
31             sword = new Item(new string[] { "sword" }, "sword", "Short range weapon!");
32             ak47 = new Item(new string[] { "ak47" }, "ak47", "Average range weapon with high damage!");
33
34             player.Inventory.Put(sword);
35             player.Inventory.Put(ak47);
36
37             grenade = new Item(new string[] { "grenade" }, "grenade",
38                               "Extreme damage and short range weapon!");
39             Bag bag1 = new Bag(new string[] { "bag1" }, "bag1", "");
40             bag1.Inventory.Put(grenade);
41             player.Inventory.Put(bag1);
42
43             LookCommand player_command = new LookCommand();
44             while (true)
45             {
46                 Console.WriteLine("Command -> ");
47                 command = Console.ReadLine();
```

```
45             string message = player_command.Execute(player, new string [] { command });
46             Console.WriteLine(message);
47         }
48     }
49 }
50 }
```

# Screenshot of Test Passing

The screenshot shows a Visual Studio IDE interface with the following components:

- Top Bar:** File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help.
- Search Bar:** Search dropdown.
- Solution Explorer:** Shows the solution 'Iteration1' with 8 projects: BagUnitTests, InventoryUnitTests, ItemUnitTests, Iteration1 (selected), LookCommandUnitTests, PlayerUnitTests, TestLocations, and UnitTest1.cs.
- Test Explorer:** Displays test results for 'Iteration1'.
  - Group Summary:** 37 Tests (37 Passed, 0 Failed, 0 Skipped) run in 149 ms.
  - Outcomes:** 5 Passed.
- Code Editor:** The 'Bag.cs' file is open, showing the implementation of the `Bag` class which implements `IHaveInventory`. It includes methods for adding items to the inventory and returning the full description of the bag.
- Status Bar:** Shows the current line (Ln: 24), character (Ch: 1), and other keyboard shortcuts like SPC and CRLF.
- Output Window:** Shows build logs:

```
Show output from: Build
Build started at 12:11 PM...
===== Build: 0 succeeded, 0 failed, 8 up-to-date
===== Build completed at 12:11 PM and took 00:05
```
- Taskbar:** Standard Windows taskbar with icons for File Explorer, Task View, Start, and others.

# Output Screenshot