```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace IdentifiableObject
8  {
9      public class Bag:Item, IHaveInventory
10     {
11         private Inventory _inventory;
12
13         public Bag(string[] ids, string name, string desc): base(ids, name,
               desc)
14         {
15             _inventory = new Inventory();
16         }
17
18         public GameObject Locate(string id)
19         {
20             if(AreYou(id)==true)
21             {
22                 return this;
23             }
24             else
25             {
26                 return _inventory.Fetch(id);
27             }
28             return null;
29         }
30
31         public string FullDescription
32         {
33             get
34             {
35                 return $"In the {this.Name} you can see:\n
                       {_inventory.ItemList}";
36             }
37         }
38
39         public Inventory Inventory
40         {
41             get
42             {
43                 return _inventory;
44             }
45         }
46     }
47 }
```

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

```
48  }
49
```



```
 1  using System;
 2  using System.Collections.Generic;
 3  using System.Linq;
 4  using System.Text;
 5  using System.Threading.Tasks;
 6
 7  namespace IdentifiableObject
 8  {
 9      public interface IHaveInventory
10      {
11          public GameObject Locate(string id);
12
13          public string Name
14          {
15              get;
16          }
17      }
18  }
19
```



```
 1  using System;
 2  using System.Collections.Generic;
 3  using System.Linq;
 4  using System.Text;
 5  using System.Threading.Tasks;
 6
 7  namespace IdentifiableObject
 8  {
 9      public abstract class Command:IdentifiableObject
10      {
11          public Command(string[] ids):base(ids)
12          {
13
14          }
15
16          public abstract string Execute(Player p, string[] text);
17      }
18  }
19
```

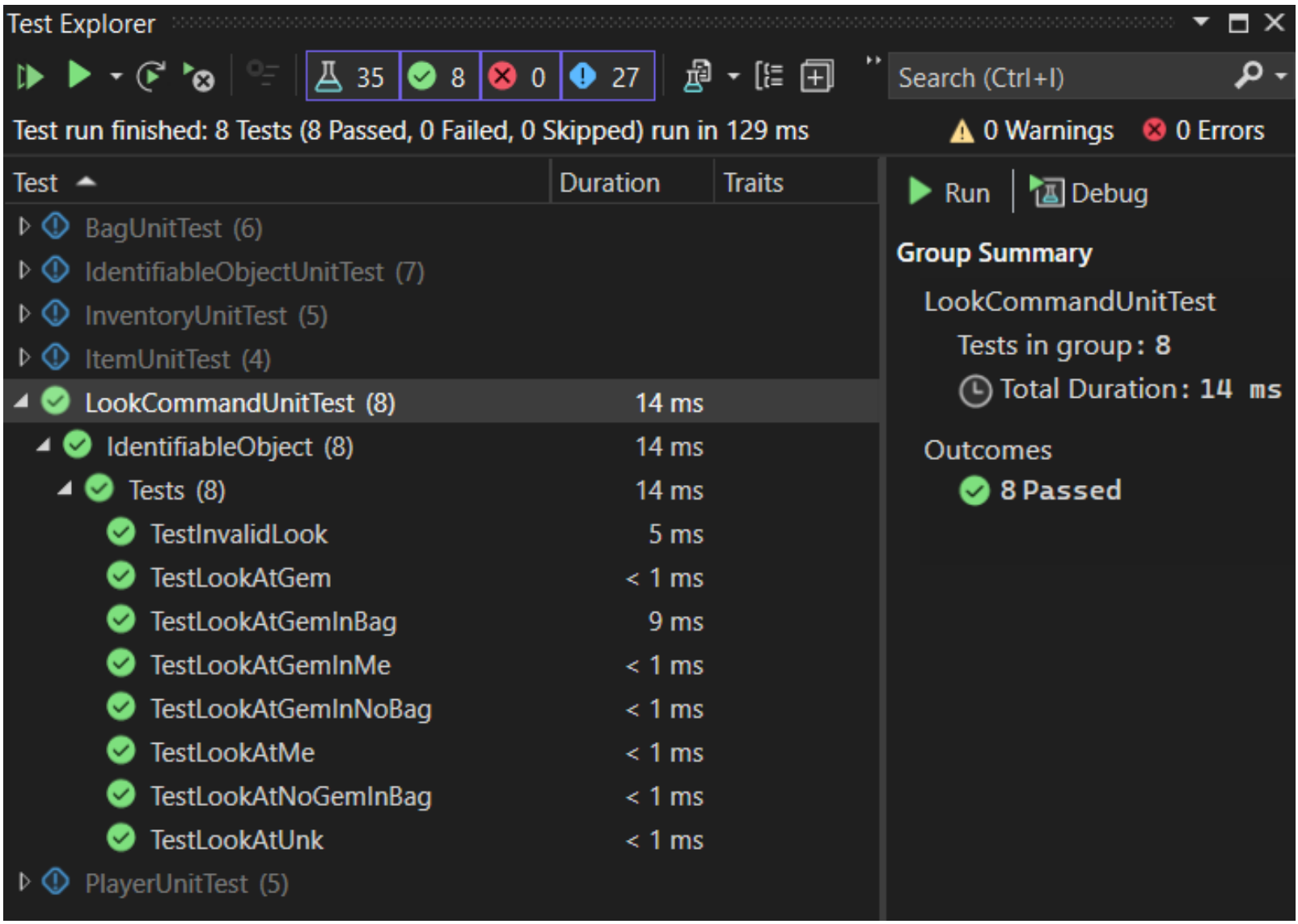
```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace IdentifiableObject
8  {
9      public class LookCommand:Command
10     {
11         public LookCommand():base(new string[] {"look"})
12         {
13
14         }
15
16         public override string Execute(Player p, string[] text)
17         {
18             if(text.Length==3 || text.Length==5)
19             {
20                 if (text[0].ToLower() == "look")
21                 {
22                     if (text[1].ToLower() == "at")
23                     {
24                         if(text.Length==3)
25                         {
26                             return LookAtIn(text[2], p);
27                         }
28                         else if(text.Length==5)
29                         {
30                             if (text[3].ToLower() == "in")
31                             {
32                                 return LookAtIn(text[2], FetchContainer(p,
                       text[4]));
33                             }
34                             else
35                             {
36                                 return "What do you want to look in?";
37                             }
38                         }
39                         else
40                         {
41                             return "I don't know how to look like that";
42                         }
43                     }
44                     else return "What do you want to look at?";
45                 }
46                 else
47                 {
48                     return "Error in look input";
```

```
49                  }
50              }
51              else
52              {
53                  return "I don't know how to look like that";
54              }
55          }
56
57          private IHaveInventory? FetchContainer(Player p, string
              containerId)
58          {
59              return p.Locate(containerId) as IHaveInventory;
60          }
61
62          private string LookAtIn(string thingId, IHaveInventory container)
63          {
64              if(container.Locate(thingId)!=null)
65              {
66                  return container.Locate(thingId).FullDescription;
67              }
68              return "I can't find the " + thingId;
69          }
70      }
71  }
72
```

```
1  namespace IdentifiableObject
2  {
3      public class Tests
4      {
5          private Player _player;
6          private Player _player_no_bag;
7          private Item _gem;
8          private Bag _bag;
9          private Command _look;
10
11         [SetUp]
12         public void Setup()
13         {
14             _look = new LookCommand();
15             _player = new Player("Duc Thang", "Student");
16             _player_no_bag = new Player("player", "participant");
17             _gem = new Item(new string[] { "gem" }, "a gem", "This is a
                 gem");
18             _bag = new Bag(new string[] { "bag" }, "Thang's bag",
                 "student");
19             _player.Inventory.Put(_bag);
20         }
21
22         [Test]
23         public void TestLookAtMe()
24         {
25             string look_execution = _look.Execute(_player, new string[]
                 { "look", "at", "inventory" });
26             string output = _player.FullDescription;
27             Assert.AreEqual(look_execution, output);
28         }
29
30         [Test]
31         public void TestLookAtGem()
32         {
33             _player.Inventory.Put(_gem);
34             string look_execution = _look.Execute(_player, new string[]
                 {"look", "at", "gem"});
35             string output = _gem.FullDescription;
36             Assert.AreEqual(look_execution, output);
37         }
38
39         [Test]
40         public void TestLookAtUnk()
41         {
42             string look_execution = _look.Execute(_player, new string[]
                 { "look", "at", "gem" });
43             string output = "I can't find the gem";
44             Assert.AreEqual(look_execution, output);
```

```
45          }
46
47          [Test]
48          public void TestLookAtGemInMe()
49          {
50              _player.Inventory.Put(_gem);
51              string look_execution = _look.Execute(_player, new string[]
                  { "look", "at", "gem", "in", "me" });
52              string output = _gem.FullDescription;
53              Assert.AreEqual(look_execution, output);
54          }
55
56          [Test]
57          public void TestLookAtGemInBag()
58          {
59              _bag.Inventory.Put(_gem);
60              string look_execution = _look.Execute(_player, new string[]
                  { "look", "at", "gem", "in", "bag" });
61              string output = _gem.FullDescription;
62              Assert.AreEqual(look_execution, output);
63          }
64
65          [Test]
66          public void TestLookAtGemInNoBag()
67          {
68              string look_execution = _look.Execute(_player_no_bag, new
                  string[] { "look", "at", "bag" });
69              string output = "I can't find the bag";
70              Assert.AreEqual(look_execution, output);
71          }
72
73          [Test]
74          public void TestLookAtNoGemInBag()
75          {
76              string look_execution = _look.Execute(_player, new string[]
                  { "look", "at", "gem", "in", "bag" });
77              string output = "I can't find the gem";
78              Assert.AreEqual(look_execution, output);
79          }
80
81          [Test]
82          public void TestInvalidLook()
83          {
84              Assert.AreEqual(_look.Execute(_player, new string[] { "look",
                  "around" }), "I don't know how to look like that");
85              Assert.AreEqual(_look.Execute(_player, new string[] { "hello",
                  "104776473" }), "I don't know how to look like that");
86              Assert.AreEqual(_look.Execute(_player, new string[] { "look",
                  "at", "Nguyen Duc Thang" }), "I can't find the Nguyen Duc
```

```
                Thang");
87          }
88      }
89  }
```



For the fix i have added the "?" after the method name "IHaveInventory" to notice the program that the return value can be null, then the second null check is the "as IHaveInventory" doesn't return the IHaveInventory type then it will return null instead.