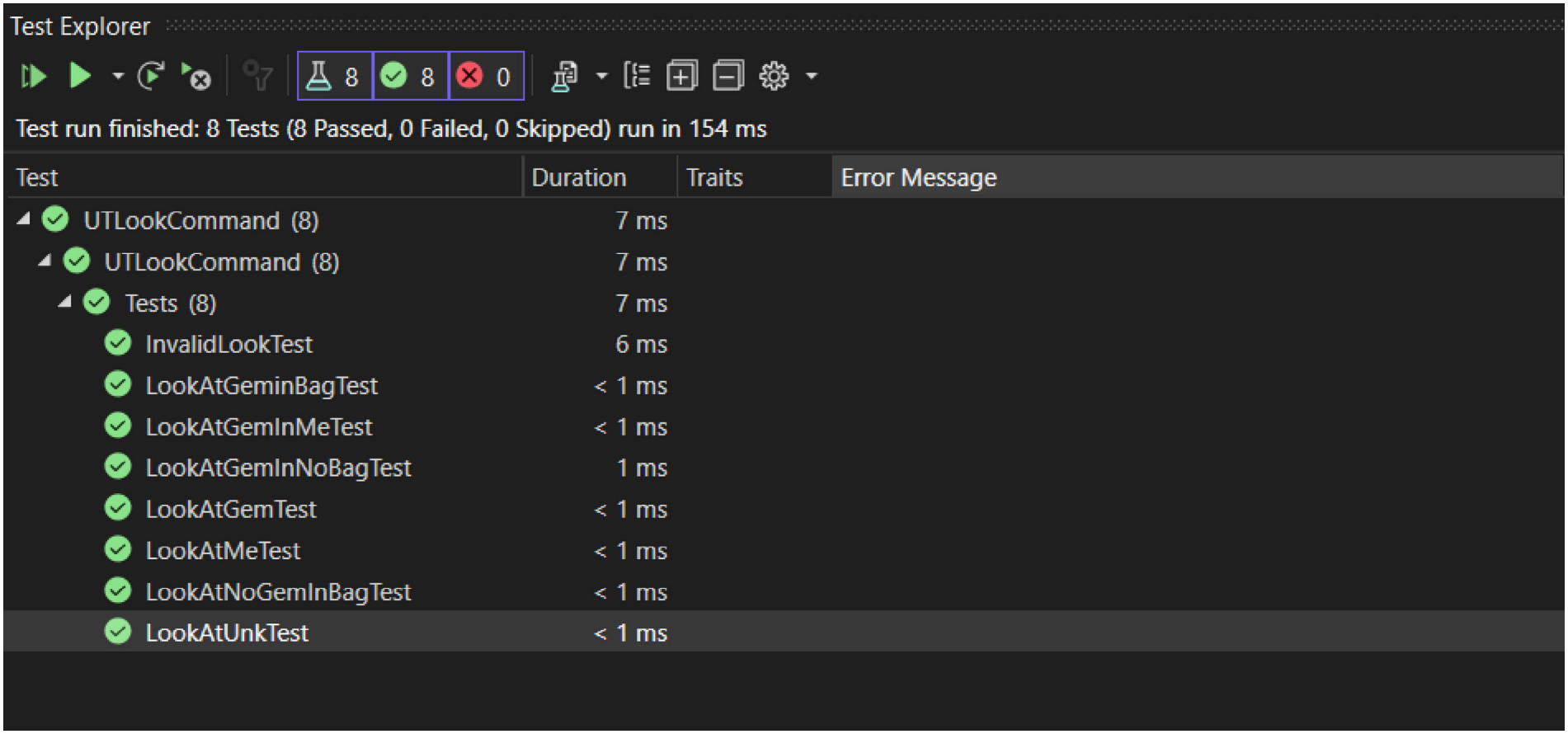
Test Explorer
8
8
0
Test run finished: 8 Tests (8 Passed, 0 Failed, 0 Skipped) run in 154 ms
Test
Duration
Traits
Error Message
UTLookCommand (8)
7 ms
UTLookCommand (8)
7 ms
Tests (8)
7 ms
InvalidLookTest
6 ms
LookAtGeminBagTest
< 1 ms
LookAtGemInMeTest
< 1 ms
LookAtGemInNoBagTest
1 ms
LookAtGemTest
< 1 ms
LookAtMeTest
< 1 ms
LookAtNoGemInBagTest
< 1 ms
LookAtUnkTest
< 1 ms

```
 1  using System;
 2  using System.Collections.Generic;
 3  using System.Linq;
 4  using System.Text;
 5  using System.Threading.Tasks;
 6
 7  namespace Iteration4
 8  {
 9      public class Bag : Item, IHaveInventory
10      {
11          private Inventory _inventory;
12          public Bag(string[] ids, string name,string desc):base(ids, name,
              desc)
13          {
14              _inventory = new Inventory();
15          }
16          public GameObject Locate(string id)
17          {
18                  if (AreYou(id))
19                  {
20                      return this;
21                  }
22                  else
23                  {
24                      return _inventory.Fetch(id);
25                  }
26              }
27          public override string FullDescription
28          {
29              get
30              {
31                  return $"In the {this.name} you can see:\n" +
                      _inventory.ItemList;
32
33              }
34          }
35
36          public Inventory Inventory
37          {
38              get { return _inventory; }
39          }
40      }
41  }
42
```

```
 1 using System;
 2 using System.Collections.Generic;
 3 using System.Linq;
 4 using System.Text;
 5 using System.Threading.Tasks;
 6
 7 namespace Iteration4
 8 {
 9     public abstract class Command : IdentifiableObject
10     {
11         public Command(string[] ids) : base(ids)
12         {
13
14         }
15         public abstract string Execute(Player p, string[] text);
16     }
17 }
18
```

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace Iteration4
8  {
9      public class GameObject : IdentifiableObject
10     {
11         private string _description;
12         private string _name;
13         public GameObject(string[] ids, string name, string desc) : base
             (ids)
14         {
15             _description = desc;
16             _name = name;
17         }
18         public string name
19         {
20             get { return _name; }
21         }
22         public string ShortDescription
23         {
24             get { return $"{_name} ({FirstID})"; }
25         }
26         public virtual string FullDescription
27         {
28             get { return _description; }
29         }
30     }
31
32 }
33
```

```
 1  using System;
 2  using System.Collections.Generic;
 3  using System.Linq;
 4  using System.Text;
 5  using System.Threading.Tasks;
 6
 7  namespace Iteration4
 8  {
 9      public interface IHaveInventory
10      {
11          public string name
12          {
13              get;
14          }
15          public GameObject Locate(string id);
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 Iteration4
 8  {
 9      public class Inventory
10      {
11          private List<Item> _items = new List<Item>();
12          public Inventory()
13          {
14
15          }
16          public bool HasItem(string id)
17          {
18              foreach (var item in _items)
19              {
20                  if (item.AreYou(id))
21                  {
22                      return true;
23                  }
24              }
25              return false;
26          }
27          public void Put(Item i)
28          {
29              _items.Add(i);
30          }
31          public Item Take(string id)
32          {
33              foreach (var item in _items)
34              {
35                  if (item.AreYou(id))
36                  {
37                      _items.Remove(item);
38                      return item;
39                  }
40
41              }
42              return null;
43          }
44          public Item Fetch(string id)
45          {
46              foreach (var item in _items)
47              {
48                  if (item.AreYou(id))
49                  {
```

```
50                      return item;
51
52                  }
53              }
54              return null;
55          }
56          public string ItemList
57          {
58              get
59              {
60                  string listItem = "";
61                  foreach (Item i in _items)
62                  {
63                      listItem = listItem + i.ShortDescription + "\n";
64                  }
65                  return listItem;
66              }
67          }
68      }
69  }
70
71
72
73
```

```
 1  using System;
 2  using System.Collections.Generic;
 3  using System.Linq;
 4  using System.Text;
 5  using System.Threading.Tasks;
 6
 7  namespace Iteration4
 8  {
 9      public class IdentifiableObject
10      {
11          private List<string> _identifiers;
12
13          public IdentifiableObject(string[] idents)
14          {
15              _identifiers = new List<string>();
16              foreach (string ident in idents)
17              {
18                  _identifiers.Add(ident.ToLower());
19              }
20
21
22          }
23
24          public bool AreYou(string name)
25          {
26              foreach (string idents in _identifiers)
27              {
28                  if (idents.ToLower() == name.ToLower())
29                  {
30                      return true;
31                  }
32              }
33
34              return false;
35          }
36
37          public string FirstID
38          {
39              get
40              {
41                  if (_identifiers.Count == 0)
42                  {
43                      return "";
44                  }
45                  else
46                  {
47                      return _identifiers.First();
48
49                  }
```

```
50              }
51          }
52
53          public void AddIdentifier(string id)
54          {
55              _identifiers.Add(id.ToLower());
56          }
57      }
58  }
59
```

```
 1  using System;
 2  using System.Collections.Generic;
 3  using System.Linq;
 4  using System.Text;
 5  using System.Threading.Tasks;
 6
 7  namespace Iteration4
 8  {
 9      public class Item : GameObject
10      {
11          public Item(string[] idents, string name, string desc) : base
              (idents, name, desc)
12          {
13
14          }
15      }
16  }
17
18
```

```
1
2  namespace Iteration4
3  {
4      public class LookCommand : Command
5      {
6          public LookCommand() : base(new string[] { "look" })
7          {
8          }
9
10         public override string Execute(Player p, string[] text)
11         {
12             if ((text.Length != 3) && (text.Length != 5))
13             {
14                 return "I don't know how to look like that";
15             }
16             else if (text[0] != "look")
17             {
18                 return "Error in look input";
19             }
20             else if (text[1] != "at")
21             {
22                 return "What do you want to look at?";
23             }
24
25             if ((text.Length == 5) && (text[3] != "in"))
26             {
27                 return "What do you want to look in?";
28             }
29
30             String itemId = text[2];
31             IHaveInventory container = p;
32
33             if (text.Length == 5)
34             {
35                 container = FetchContainer(p, text[4]);
36                 if (container == null)
37                 {
38                     return $"I cannot find the {text[4]}";
39                 }
40             }
41
42             return LookAtIn(itemId, container);
43         }
44
45         private IHaveInventory FetchContainer(Player p, string containerId)
46         {
47             return p.Locate(containerId) as IHaveInventory;
48         }
49
```

```
50          private string LookAtIn(string thingId, IHaveInventory container)
51          {
52              var item = container.Locate(thingId);
53              if (item != null)
54              {
55                  return item.FullDescription;
56              }
57              else
58              {
59                  return $"I can't find the {thingId}";
60              }
61          }
62      }
63  }
64
```

```
 1  using System;
 2  using System.Collections.Generic;
 3  using System.Linq;
 4  using System.Text;
 5  using System.Threading.Tasks;
 6
 7  namespace Iteration4
 8  {
 9      public class Player : GameObject, IHaveInventory
10      {
11          private Inventory _inventory;
12          public Player(string name, string desc) : base(new string[] { "me",
               "inventory" }, name, desc)
13          {
14              _inventory = new Inventory();
15          }
16          public GameObject Locate(string id)
17          {
18              if (AreYou(id))
19              {
20                  return this;
21              }
22              else
23              {
24                  return _inventory.Fetch(id);
25              }
26          }
27          public override string FullDescription
28          {
29              get
30              {
31                  return $"You are {this.name}. You are carrying:\n" +
                      _inventory.ItemList;
32              }
33          }
34          public Inventory Inventory
35          {
36              get
37              {
38                  return _inventory;
39              }
40          }
41
42      }
43  }
44
```

```
1  namespace Iteration4
2  {
3      internal class Program
4      {
5          static void Main(string[] args)
6          {
7              Console.WriteLine("Hello, World!");
8          }
9      }
10 }
11
```

```
 1  using Iteration4;
 2  using NUnit.Framework;
 3
 4  namespace UTLookCommand
 5  {
 6      public class Tests
 7      {
 8          private Player _player;
 9          private Item gem, spade;
10          private Bag bag;
11          private LookCommand lookCommand;
12
13          [SetUp]
14          public void Setup()
15          {
16              _player = new Player("Player1", "first player");
17              gem = new Item(new string[] { "gem", "a gem" }, "purple gem",
                  "big purple gem");
18              spade = new Item(new string[] { "spade" }, "purple spade",
                  "big purple spade");
19
20              lookCommand = new LookCommand();
21              bag = new Bag(new string[] { "testBag" }, "a bag", "contains
                  items");
22          }
23
24          [Test]
25          public void LookAtMeTest()
26          {
27              _player.Inventory.Put(gem);
28              var expectedOutcome = _player.FullDescription;
29              string desc = lookCommand.Execute(_player, new string[]
                  { "look", "at", "inventory" });
30              Assert.That(desc, Is.EqualTo(expectedOutcome));
31          }
32
33          [Test]
34          public void LookAtGemTest()
35          {
36              _player.Inventory.Put(gem);
37              var expectedOutcome = gem.FullDescription;
38              var result = lookCommand.Execute(_player, new string[]
                  { "look", "at", "gem" });
39              Assert.That(result, Is.EqualTo(expectedOutcome));
40          }
41
42          [Test]
43          public void LookAtUnkTest()
44          {
```

```
45              var expectedOutcome = "I can't find the gem";
46              var result = lookCommand.Execute(_player, new string[]
                  { "look", "at", "gem" });
47              Assert.That(result, Is.EqualTo(expectedOutcome));
48          }
49
50          [Test]
51          public void LookAtGemInMeTest()
52          {
53              _player.Inventory.Put(gem);
54              var expectedOutcome = gem.FullDescription;
55              var result = lookCommand.Execute(_player, new string[]
                  { "look", "at", "gem", "in", "me" });
56              Assert.That(result, Is.EqualTo(expectedOutcome));
57          }
58
59          [Test]
60          public void LookAtGeminBagTest()
61          {
62              _player.Inventory.Put(gem);
63              var expectedOutcome = gem.FullDescription;
64              var result = lookCommand.Execute(_player, new string[]
                  { "look", "at", "gem", "in", "Inventory" });
65              Assert.That(result, Is.EqualTo(expectedOutcome));
66          }
67
68          [Test]
69          public void LookAtGemInNoBagTest()
70          {
71              var expectedOutcome = "I can't find the bag";
72              var result = lookCommand.Execute(_player, new string[]
                  { "look", "at", "bag", "in", "me" });
73              Assert.That(result, Is.EqualTo(expectedOutcome));
74          }
75
76          [Test]
77          public void LookAtNoGemInBagTest()
78          {
79              var expectedOutcome = "I can't find the gem";
80              var result = lookCommand.Execute(_player, new string[]
                  { "look", "at", "gem", "in", "me" });
81              Assert.That(result, Is.EqualTo(expectedOutcome));
82          }
83
84          [Test]
85          public void InvalidLookTest()
86          {
87              var result0 = lookCommand.Execute(_player, new string[]
                  { "look", "there" });
```

```
88              Assert.That(result0, Is.EqualTo("I don't know how to look like
                   that"));
89
90              var result1 = lookCommand.Execute(_player, new string[]
                  { "there", "it", "is" });
91              Assert.That(result1, Is.EqualTo("Error in look input"));
92
93              var result2 = lookCommand.Execute(_player, new string[]
                  { "look", "over", "there" });
94              Assert.That(result2, Is.EqualTo("What do you want to look
                  at?"));
95
96              var result3 = lookCommand.Execute(_player, new string[]
                  { "look", "at", "gem", "over", "there" });
97              Assert.That(result3, Is.EqualTo("What do you want to look
                  in?"));
98          }
99      }
100 }
```