```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace SwinAdventure
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 id in idents)
17             {
18                 _identifiers.Add(id.ToLower());
19             }
20         }
21
22         public bool AreYou(string id)
23         {
24             return _identifiers.Contains(id.ToLower());
25         }
26
27         public string FirstId
28         {
29             get
30             {
31                 if (_identifiers.Count > 0)
32                 {
33                     return _identifiers[0];
34                 }
35                 else
36                 {
37                     return "";
38                 }
39             }
40         }
41
42         public void AddIdentifier(string id)
43         {
44             _identifiers.Add(id.ToLower());
45         }
46     }
47 }
48
```

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace SwinAdventure
8  {
9      public class GameObject : IdentifiableObject
10     {
11         private string _name;
12         private string _description;
13         private string[] _ids;
14
15         public GameObject(string[] ids, string name, string desc) : base
             (ids)
16         {
17             _name = name;
18             _description = desc;
19             _ids = ids;
20         }
21
22         public string Name
23         {
24             get
25             {
26                 return _name;
27             }
28         }
29
30         public string ShortDescription
31         {
32             get
33             {
34                 return _name + " (" + _ids[0] + ")";
35             }
36         }
37
38         public virtual string FullDescription
39         {
40             get
41             {
42                 return _description;
43             }
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
7  namespace SwinAdventure
8  {
9      public class Inventory
10     {
11         List<Item> _items;
12
13         public Inventory()
14         {
15             _items = new List<Item>();
16         }
17
18         public bool HasItem(string id)
19         {
20             foreach (Item i in _items)
21             {
22                 if (i.AreYou(id))
23                 {
24                     return true;
25                 }
26             }
27             return false;
28         }
29
30         public void Put(Item itm)
31         {
32             _items.Add(itm);
33         }
34
35         public Item? Take(string id)
36         {
37             Item? t = null;
38             foreach (Item i in _items)
39             {
40                 if (i.AreYou(id))
41                 {
42                     t = i;
43                     _items.Remove(i);
44                     return t;
45                 }
46             }
47             return t;
48         }
49
```

```
50          public Item? Fetch(string id)
51          {
52              Item? t = null;
53              foreach (Item i in _items)
54              {
55                  if (i.AreYou(id))
56                  {
57                      return i;
58                  }
59              }
60              return t;
61          }
62
63          public string ItemList
64          {
65              get
66              {
67                  string list = "";
68                  foreach (Item i in _items)
69                  {
70                      list += "\t" + i.ShortDescription + "\n";
71                  }
72                  return list;
73              }
74          }
75      }
76  }
77
```

```
 1  using System;
 2  using System.Collections.Generic;
 3  using System.Linq;
 4  using System.Text;
 5  using System.Threading.Tasks;
 6
 7  namespace SwinAdventure
 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
```

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

```
1  using SwinAdventure;
2
3  namespace IdentifiableObjectTest
4  {
5      public class IdentifiableObjectTest
6      {
7          [SetUp]
8          public void Setup()
9          {
10
11         }
12
13         [Test]
14         public void TestAreYou()
15         {
16             IdentifiableObject obj = new IdentifiableObject(new string[]
                 {"fred", "bob"});
17
18             bool result1 = obj.AreYou("fred");
19             bool result2 = obj.AreYou("bob");
20
21             Assert.IsTrue(result1);
22             Assert.IsTrue(result2);
23         }
24
25         [Test]
26         public void TestNotAreYou()
27         {
28             IdentifiableObject obj = new IdentifiableObject(new string[]
                 { "fred", "bob" });
29
30             bool result1 = obj.AreYou("wilma");
31             bool result2 = obj.AreYou("boby");
32
33             Assert.IsFalse(result1);
34             Assert.IsFalse(result2);
35         }
36
37         [Test]
38         public void TestCaseSensitive()
39         {
40             IdentifiableObject obj = new IdentifiableObject(new string[]
                 { "fred", "bob" });
41
42             bool result1 = obj.AreYou("FRED");
43             bool result2 = obj.AreYou("bOB");
44
45             Assert.IsTrue(result1);
46             Assert.IsTrue(result2);
```

```
47          }
48
49          [Test]
50          public void TestFirstID()
51          {
52              IdentifiableObject obj = new IdentifiableObject(new string[]
                { "fred", "bob" });
53
54              string firstID = obj.FirstId;
55
56              Assert.AreEqual("fred", firstID);
57          }
58
59          [Test]
60          public void TestFirstIDWithNoIDs()
61          {
62              IdentifiableObject obj = new IdentifiableObject(new string[]
                { });
63
64              string firstID = obj.FirstId;
65
66              Assert.AreEqual("", firstID);
67          }
68
69          [Test]
70          public void TestAddIdentifier()
71          {
72              IdentifiableObject obj = new IdentifiableObject(new string[]
                { "fred", "bob" });
73
74              obj.AddIdentifier("wilma");
75
76              Assert.IsTrue(obj.AreYou("fred"));
77              Assert.IsTrue(obj.AreYou("bob"));
78              Assert.IsTrue(obj.AreYou("wilma"));
79          }
80      }
81  }
```

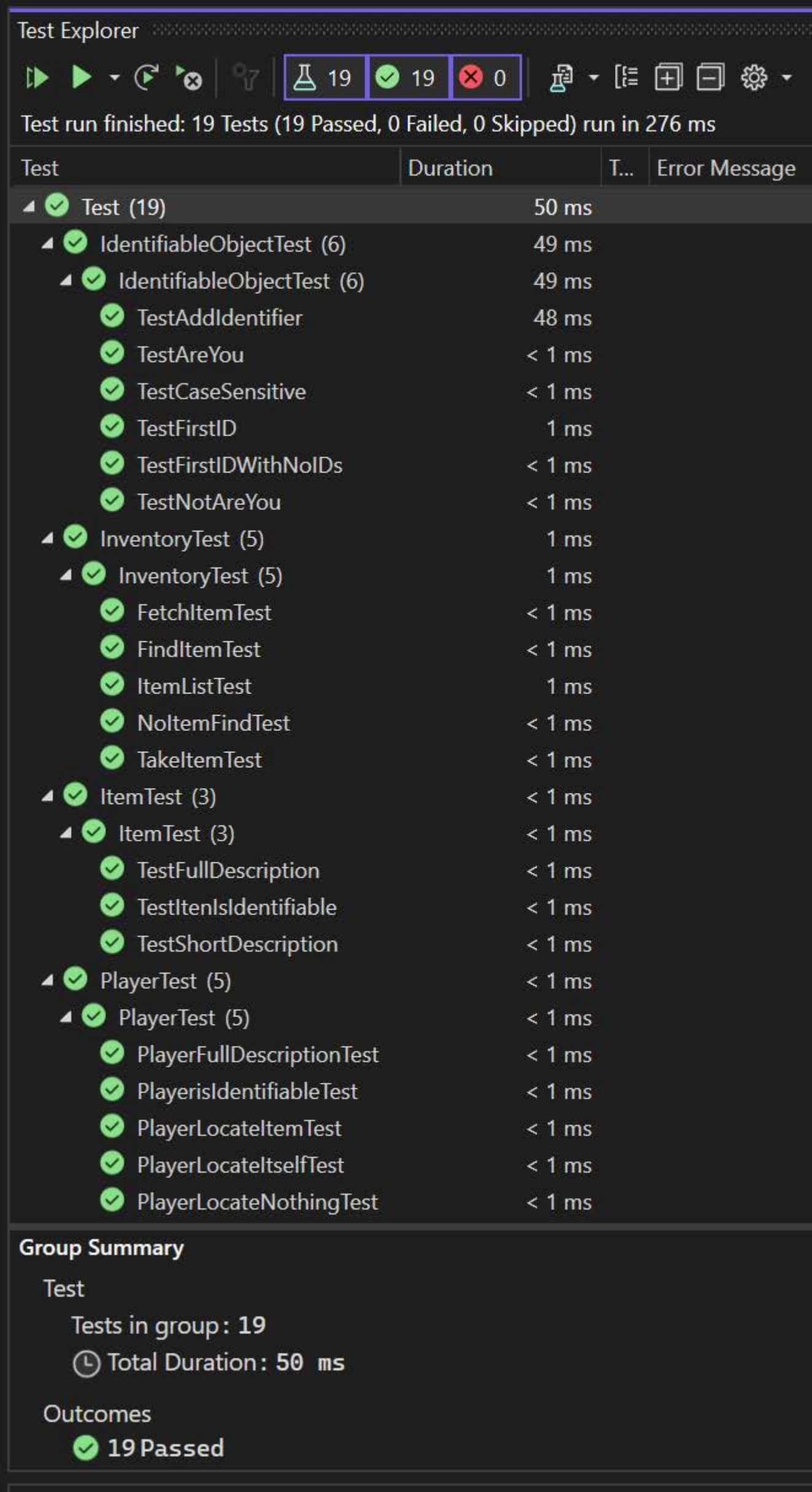
```
1  using SwinAdventure;
2  using System;
3  using System.Collections.Generic;
4  using System.Linq;
5  using System.Text;
6  using System.Threading.Tasks;
7
8  namespace InventoryTest
9  {
10     internal class InventoryTest
11     {
12         Item BronzeSword = new Item(new string[] { "sword", "weapon" },
             "Bronze Sword", "A simple bronze sword.");
13         Item BronzeAxe = new Item(new string[] { "axe", "weapon" }, "Bronze
              Axe", "A simple bronze axe.");
14         [SetUp]
15         public void Setup()
16         {
17
18         }
19
20         [Test]
21         public void FindItemTest()
22         {
23             Inventory inv = new Inventory();
24
25             inv.Put(BronzeAxe);
26             inv.Put(BronzeSword);
27
28             bool sword = inv.HasItem("sword");
29             bool axe = inv.HasItem("axe");
30
31             Assert.IsTrue(sword);
32             Assert.IsTrue(axe);
33         }
34
35         [Test]
36         public void NoItemFindTest()
37         {
38             Inventory inv = new Inventory();
39
40             Assert.IsFalse(inv.HasItem("sword"));
41             Assert.IsFalse(inv.HasItem("axe"));
42         }
43
44         [Test]
45         public void FetchItemTest()
46         {
47             Inventory inv = new Inventory();
```

```
48
49              inv.Put(BronzeAxe);
50              inv.Put(BronzeSword);
51
52              Item sword = inv.Fetch("sword");
53              Item axe = inv.Fetch("axe");
54
55              Assert.AreEqual(BronzeSword, sword);
56              Assert.AreEqual(BronzeAxe, axe);
57          }
58
59          [Test]
60          public void TakeItemTest()
61          {
62              Inventory inv = new Inventory();
63
64              inv.Put(BronzeAxe);
65              inv.Put(BronzeSword);
66
67              Item sword = inv.Take("sword");
68              Item axe = inv.Take("axe");
69
70              Item SwordRemain = inv.Fetch("sword");
71              Item AxeRemain = inv.Fetch("axe");
72
73              Assert.AreEqual(BronzeSword, sword);
74              Assert.AreEqual(BronzeAxe, axe);
75              Assert.IsTrue(SwordRemain == null);
76              Assert.IsTrue(AxeRemain == null);
77          }
78
79          [Test]
80          public void ItemListTest()
81          {
82              Inventory inv = new Inventory();
83
84              inv.Put(BronzeAxe);
85              inv.Put(BronzeSword);
86
87              string list = inv.ItemList;
88
89              Assert.AreEqual("\tBronze Axe (axe)\n\tBronze Sword (sword)\n", ↵
                    list);
90          }
91      }
92  }
93
```

```
1  using SwinAdventure;
2
3  namespace ItemTest
4  {
5      public class ItemTest
6      {
7          Item BronzeSword = new Item(new string[] { "sword", "weapon" },
             "Bronze Sword", "A simple bronze sword.");
8          Item BronzeAxe = new Item(new string[] { "axe", "weapon" }, "Bronze
              Axe", "A simple bronze axe.");
9          [SetUp]
10         public void Setup()
11         {
12
13         }
14
15         [Test]
16         public void TestItenIsIdentifiable()
17         {
18             bool SwordisIdentifiable = BronzeSword.AreYou("sword");
19             bool AxeisIdentifiable = BronzeAxe.AreYou("sword");
20
21             Assert.IsTrue(SwordisIdentifiable);
22             Assert.IsFalse(AxeisIdentifiable);
23         }
24
25         [Test]
26         public void TestShortDescription()
27         {
28             string SwordShortDesc = BronzeSword.ShortDescription;
29             string AxeShortDesc = BronzeAxe.ShortDescription;
30
31             Assert.AreEqual("Bronze Sword (sword)", SwordShortDesc);
32             Assert.AreEqual("Bronze Axe (axe)", AxeShortDesc);
33         }
34
35         [Test]
36         public void TestFullDescription()
37         {
38             string SwordFullDesc = BronzeSword.FullDescription;
39             string AxeFullDesc = BronzeAxe.FullDescription;
40
41             Assert.AreEqual("A simple bronze sword.", SwordFullDesc);
42             Assert.AreEqual("A simple bronze axe.", AxeFullDesc);
43         }
44     }
45 }
46
```

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6  using SwinAdventure;
7
8  namespace PlayerTest
9  {
10     internal class PlayerTest
11     {
12         Player _player = new Player("Sen", "luv ur mom");
13         Item BronzeSword = new Item(new string[] { "sword", "weapon" },
             "Bronze Sword", "A simple bronze sword.");
14         [SetUp]
15         public void Setup()
16         {
17
18         }
19
20         [Test]
21         public void PlayerisIdentifiableTest()
22         {
23             Assert.IsTrue(_player.AreYou("me"));
24             Assert.IsTrue(_player.AreYou("inventory"));
25         }
26
27         [Test]
28         public void PlayerLocateItemTest()
29         {
30             _player.Inventory.Put(BronzeSword);
31
32             Assert.AreEqual(BronzeSword, _player.Locate("sword"));
33         }
34
35         [Test]
36         public void PlayerLocateItselfTest()
37         {
38             Assert.AreEqual(_player, _player.Locate("me"));
39             Assert.AreEqual(_player, _player.Locate("inventory"));
40         }
41
42         [Test]
43         public void PlayerLocateNothingTest()
44         {
45             Assert.IsNull(_player.Locate("sadaw"));
46         }
47
48         [Test]
```

```
49          public void PlayerFullDescriptionTest()
50          {
51              _player.Inventory.Put(BronzeSword);
52              string desc = _player.FullDescription;
53
54              Assert.AreEqual("You are Sen, luv ur mom. You are carrying:
                  \tBronze Sword (sword)\n", desc);
55          }
56      }
57  }
58
```


Test Explorer
19
19
0
Test run finished: 19 Tests (19 Passed, 0 Failed, 0 Skipped) run in 276 ms
Test
Duration
T...
Error Message
Test (19) 50 ms
IdentifiableObjectTest (6) 49 ms
IdentifiableObjectTest (6) 49 ms
TestAddIdentifier 48 ms
TestAreYou < 1 ms
TestCaseSensitive < 1 ms
TestFirstID 1 ms
TestFirstIDWithNoIDs < 1 ms
TestNotAreYou < 1 ms
InventoryTest (5) 1 ms
InventoryTest (5) 1 ms
FetchItemTest < 1 ms
FindItemTest < 1 ms
ItemListTest 1 ms
NoItemFindTest < 1 ms
TakeItemTest < 1 ms
ItemTest (3) < 1 ms
ItemTest (3) < 1 ms
TestFullDescription < 1 ms
TestItenIsIdentifiable < 1 ms
TestShortDescription < 1 ms
PlayerTest (5) < 1 ms
PlayerTest (5) < 1 ms
PlayerFullDescriptionTest < 1 ms
PlayerisIdentifiableTest < 1 ms
PlayerLocateItemTest < 1 ms
PlayerLocateItselfTest < 1 ms
PlayerLocateNothingTest < 1 ms
Group Summary
Test
Tests in group: 19
Total Duration: 50 ms
Outcomes
19 Passed