

[KingSchlock](#) / [UploadedCodeToPrintAgain](#) Private[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Security](#) [Insights](#) [Settings](#)[main](#) ▾

...

[UploadedCodeToPrintAgain](#) / [TestLookCommand.cs](#) / <> Jump to ▾

KingSchlock Add files via upload



1 contributor

116 lines (98 sloc) | 4.19 KB

...

```
1  using SwinAdventure;
2
3  namespace SwinAdventureTests
4  {
5      [TestFixture()]
6      public class TestLookCommand
7      {
8          LookCommand lookTest;
9          Player playerTest;
10         Bag bagTest;
11         Item swordTest;
12
13         string unknown, noBag,
14             badLength, badLook,
15             badAt, badIn;
16
17         [SetUp()]
18         public void Setup()
19         {
20             lookTest = new();
21             playerTest = new("thomas", "The mighty keyboard warrior");
22             bagTest = new(new string[] { "satchel" }, "satchel", "it's smol");
23             swordTest = new(new string[] { "sword" }, "sword", "lil poker");
24
25             unknown = "I can't find the sword";
26             noBag = $"I can't find the {bagTest.Name}";
27
28             badLength = "I don't know how to look for that.";
29             badLook = "Error in look input";
30             badAt = "What do you want to look at?";
31             badIn = "What do you want to look in?";
32
```

```
33     playerTest.Inventory.Put(swordTest);
34 }
35
36 ///! Return players descript when looking at the inventory
37 [Test()]
38 public void TestLookAtMe()
39 {
40     Assert.That(lookTest.Execute(playerTest, new string[] { "look", "at", "me" }),
41         Is.EqualTo(playerTest.FullDescription));
42 }
43
44 ///! Returns item description when looking for an item in players invent
45 [Test()]
46 public void TestLookAtItem()
47 {
48     Assert.That(lookTest.Execute(playerTest, new string[] { "look", "at", "sword" }),
49         Is.EqualTo(swordTest.FullDescription));
50 }
51
52 ///! Responds unknown when item isn't in inventory
53 [Test()]
54 public void TestLookAtUnkn()
55 {
56     playerTest.Inventory.Take("sword");
57
58     Assert.That(lookTest.Execute(playerTest, new string[] { "look", "at", "sword", "in"
59         Is.EqualTo(unknown));
60 }
61
62 ///! Returns item description when searching for item specifically in invent
63 [Test()]
64 public void TestLookAtItemInInventory()
65 {
66     Assert.That(lookTest.Execute(playerTest, new string[] { "look", "at", "sword", "in"
67         Is.EqualTo(swordTest.FullDescription));
68 }
69
70 ///! Returns item description when searching for it in a bag in players invent
71 [Test()]
72 public void TestLookAtItemInBag()
73 {
74     playerTest.Inventory.Take("sword");
75
76     bagTest.Inventory.Put(swordTest);
77     playerTest.Inventory.Put(bagTest);
78
79     Assert.That(lookTest.Execute(playerTest, new string[] { "look", "at", "sword", "in", "s
80         Is.EqualTo(swordTest.FullDescription));
81 }
82
83 ///! Returns noBag when there's no container in players invent
84 [Test()]
```

```
85 public void TestLookAtItemInNoBag()
86 {
87     Assert.That(lookTest.Execute(playerTest, new string[] { "look", "at", "sword", "in"
88         Is.EqualTo(noBag));
89 }
90
91 ///! Returns unknown when requested item isn't in bag
92 [Test()]
93 public void TestLookAtNoItemInBag()
94 {
95     playerTest.Inventory.Put(bagTest);
96
97     Assert.That(lookTest.Execute(playerTest, new string[] { "look", "at", "sword", "in"
98         Is.EqualTo(unknown));
99 }
100
101 ///! Tests all error conditions
102 public void TestInvalidLook(string look, string result)
103 {
104     Assert.Multiple(() => {
105         Assert.That(lookTest.Execute(playerTest, new string[] { "aaaaa" }),
106             Is.EqualTo(badLength));
107         Assert.That(lookTest.Execute(playerTest, new string[] { "search", "at", "sword"
108             Is.EqualTo(badLook));
109         Assert.That(lookTest.Execute(playerTest, new string[] { "look", "for", "sword"
110             Is.EqualTo(badAt));
111         Assert.That(lookTest.Execute(playerTest, new string[] { "look", "for", "sword"
112             Is.EqualTo(badIn));
113     }); ///? can i use testcases here?
114 }
115 }
116 }
```