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

```
using SwinAdventure;

namespace SwinAdventureTests
{
    [TestFixture()]
    public class TestLookCommand
    {
        LookCommand lookTest;
        Player playerTest;
        Bag bagTest;
        Item swordTest;

        string unknown, noBag,
            badLength, badLook,
            badAt, badIn;

        [SetUp()]
        public void Setup()
        {
            lookTest = new();
            playerTest = new("thomas", "The mighty keyboard warrior");
            bagTest = new(new string[] { "satchel" }, "satchel", "it's smol");
            swordTest = new(new string[] { "sword" }, "sword", "lil poker");

            unknown = "I can't find the sword";
            noBag = $"I can't find the {bagTest.Name}";

            badLength = "I don't know how to look for that.";
            badLook = "Error in look input";
            badAt = "What do you want to look at?";
            badIn = "What do you want to look in?";

            playerTest.Inventory.Put(swordTest);
        }

        //! Return players descript when looking at the inventory
        [Test()]
        public void TestLookAtMe()
        {
            Assert.That(lookTest.Execute(playerTest, new string[] {"look", "at", "me"}),
                Is.EqualTo(playerTest.FullDescription));
        }

        //! Returns item description when looking for an item in players invent
        [Test()]
        public void TestLookAtItem()
        {
            Assert.That(lookTest.Execute(playerTest, new string[] {"look", "at", "sword"}),
                Is.EqualTo(swordTest.FullDescription));
        }

        //! Responds unknown when item isn't in inventory
        [Test()]
        public void TestLookAtUnkn()
        {
            playerTest.Inventory.Take("sword");

            Assert.That(lookTest.Execute(playerTest, new string[] { "look", "at", "sword", "in'
                Is.EqualTo(unknown));
        }

        //! Returns item description when searching for item specifically in invent
        [Test()]
        public void TestLookAtItemInInventory()
        {
            Assert.That(lookTest.Execute(playerTest, new string[] {"look", "at", "sword", "in",
                Is.EqualTo(swordTest.FullDescription));
        }

        //! Returns item description when searching for it in a bag in players invent
        [Test()]
        public void TestLookAtItemInBag()
        {
            playerTest.Inventory.Take("sword");

            bagTest.Inventory.Put(swordTest);
            playerTest.Inventory.Put(bagTest);

            Assert.That(lookTest.Execute(playerTest, new string[] {"look","at","sword","in","s
                Is.EqualTo(swordTest.FullDescription));
        }

        //! Returns noBag when there's no container in players invent
        [Test()]
        public void TestLookAtItemInNoBag()
        {
            Assert.That(lookTest.Execute(playerTest, new string[] { "look", "at", "sword", "in'
                Is.EqualTo(noBag));
        }

        //! Returns unknown when requested item isn't in bag
        [Test()]
        public void TestLookAtNoItemInBag()
        {
            playerTest.Inventory.Put(bagTest);

            Assert.That(lookTest.Execute(playerTest, new string[] { "look", "at", "sword", "in'
                Is.EqualTo(unknown));
        }

        //! Tests all error conditions
        public void TestInvalidLook(string look, string result)
        {
            Assert.Multiple(() => {
                Assert.That(lookTest.Execute(playerTest, new string[] {"aaaaa"}),
                    Is.EqualTo(badLength));
                Assert.That(lookTest.Execute(playerTest, new string[] { "search", "at", "sword'
                    Is.EqualTo(badLook));
                Assert.That(lookTest.Execute(playerTest, new string[] { "look", "for", "sword",
                    Is.EqualTo(badAt));
                Assert.That(lookTest.Execute(playerTest, new string[] { "look", "for", "sword",
                    Is.EqualTo(badIn));
            }); //? can i use testcases here?
        }
    }
}
```