Wyższa Szkoła Bankowa

Programowanie Obiektowe

Ćwiczenia 2 - zadania

Imiona I nazwiska: Artem Pushkarov, Maksym Cherniakov
Adresy mail: furyX66@gmail.com, mchernyako2003@gmail.com
Id's: 144819, 144806
Link na git:
https://github.com/furyX66/ProgramowanieObiektowe_Zadanie2_2023

Rozdział 1 – Zadanie 1

1. Dodać klasę ScreenLineEntry o następujących polach:
   - BackgroundColor
   - ForegroundColor
   - Text

```
7 references
public class ScreenLineEntry
{
    6 references | 2/2 passing
    public ConsoleColor BackgroundColor { get; set; }
    6 references | 2/2 passing
    public ConsoleColor ForegroundColor { get; set; }
    6 references | 2/2 passing
    public string? Text { get; set; }

}
```

2. Dodać klasę o nazwie ScreenDefinition o następujących polach:
   - LineEntries - typu List<ScreenLineEntry>

```
22 references
public class ScreenDefinition
{
    99+ references | 1/1 passing
    public List<ScreenLineEntry> Screens { get; set; }
}
```

3. Dodać klasę statyczną o nazwie ScreenDefinionService , która zawiera następujące metody
   - ScreenDefinition Load(string jsonFileName)
   - bool Save(ScreenDefinition screenDefinition, string jsonFileName)

```
99+ references
public static class ScreenDefinitionService
{
    8 references | 1/1 passing
    public static ScreenDefinition Load(string jsonFileName)
    {
        if (File.Exists(jsonFileName))
        {
            string? json = File.ReadAllText(jsonFileName);
            return JsonConvert.DeserializeObject<ScreenDefinition>(json);
        }
        else
        {
            throw new FileNotFoundException("JSON file not found.", jsonFileName);
        }
    }

    1 reference | 1/1 passing
    public static void Save(ScreenDefinition screenDefinition, string jsonFileName)
    {
        string json = JsonConvert.SerializeObject(screenDefinition, Formatting.Indented);
        File.WriteAllText(jsonFileName, json);
    }
    99+ references | 1/1 passing
    public static void PrintLine(List<ScreenLineEntry> lines, int lineNumber)
    {
        Console.ForegroundColor = lines[lineNumber].ForegroundColor;
        Console.BackgroundColor = lines[lineNumber].BackgroundColor;
        Console.WriteLine(lines[lineNumber].Text);
        Console.ResetColor();
    }
}
```

4. Dodać nowy projekt o nazwie SampleHierarchies.Services.Tests, do którego należy dodać unit testy klasy ScreenDefinitionService.cs

```
[TestMethod]
0 references
public void Load_NonExistentJsonFile_ThrowsFileNotFoundExceptionAndExits()
{
    // Arrange
    string jsonFileName = "nonexistent.json";

    // Act & Assert
    try
    {
        ScreenDefinition result = ScreenDefinitionService.Load(jsonFileName);
        Assert.Fail("Expected exception was not thrown");
    }
    catch (FileNotFoundException ex)
    {
        Assert.AreEqual("JSON file not found.", ex.Message);
    }
}
```

```
[TestClass]
0 references
public class ScreenDefinitionServiceTests
{
    [TestMethod]
    0 references
    public void Load_ValidJsonFile_ReturnsScreenDefinition()
    {
        // Arrange
        var lines = new List<ScreenLineEntry>
    {
        new ScreenLineEntry
        {
            BackgroundColor = ConsoleColor.DarkBlue,
            ForegroundColor = ConsoleColor.White,
            Text = "Test line"
        }
    };
        int lineNumber = 0;
        string expectedLine = "Test line";

        // Act
        using (StringWriter sw = new StringWriter())
        {
            Console.SetOut(sw);
            ScreenDefinitionService.PrintLine(lines, lineNumber);
            string printedLine = sw.ToString().Trim();

            // Assert
            Assert.AreEqual(expectedLine, printedLine);
        }
    }
```

```
public void Save_WritesExpectedJsonToFile()
{
    // Arrange
    string jsonFileName = "output.json";
    var screenDefinition = new ScreenDefinition
    {
        Screens = new List<ScreenLineEntry>
    {
        new ScreenLineEntry
        {
            BackgroundColor = ConsoleColor.DarkBlue,
            ForegroundColor = ConsoleColor.White,
            Text = "Line 1"
        },
        new ScreenLineEntry
        {
            BackgroundColor = ConsoleColor.Black,
            ForegroundColor = ConsoleColor.Green,
            Text = "Line 2"
        }
    }
    };

    // Act
    ScreenDefinitionService.Save(screenDefinition, jsonFileName);

    // Assert
    string savedJson = File.ReadAllText(jsonFileName);
    var deserializedScreenDefinition = JsonConvert.DeserializeObject<ScreenDefinition>(savedJson);

    Assert.IsNotNull(deserializedScreenDefinition);
    Assert.AreEqual(screenDefinition.Screens.Count, deserializedScreenDefinition.Screens.Count);

    for (int i = 0; i < screenDefinition.Screens.Count; i++)
    {
        Assert.AreEqual(screenDefinition.Screens[i].BackgroundColor, deserializedScreenDefinition.Screens[i].BackgroundColor);
        Assert.AreEqual(screenDefinition.Screens[i].ForegroundColor, deserializedScreenDefinition.Screens[i].ForegroundColor);
        Assert.AreEqual(screenDefinition.Screens[i].Text, deserializedScreenDefinition.Screens[i].Text);
    }

    File.Delete(jsonFileName);
}
```

5. Dodać do klasy Screen.cs pole o nazwie ScreenDefinitionJson, które będzie przechowywało nazwę pliku z definicją ekranu.

```
public abstract class Screen
{
    #region Class Members
    0 references
    public ConsoleColor consoleColor { get; set; }
    14 references
    public virtual string? ScreenDefinitionJson { get; set; }
```

Przykład nadpisania tej właściwości w klasie MainScreen

```
public sealed class MainScreen : Screen
{
    #region Properties And Ctor

    /// <summary>
    /// Data service.
    /// </summary>
    private IDataService _dataService;
    private ScreenDefinition _screenDefinition;

    public override string? ScreenDefinitionJson { get; set; } = "MainScreen.json";
```

Pozostałe ekrany są wykonywane w ten sam sposób.

6. Dodać pliki JSON dla każdego z ekranów z definicją ich zawartości.

Przykładowy plik dla każdego ekranu

```
"Screens": [
  {
    "BackgroundColor": 15,
    "ForegroundColor": 0,
    "Text": "MainSceen" //0
  },
  {
    "BackgroundColor": 0,
    "ForegroundColor": 15,
    "Text": "Your available choices are:"//1
  },
  {
    "BackgroundColor": 0,
    "ForegroundColor": 15,
    "Text": "0. Exit" //2
  },
  {
    "BackgroundColor": 0,
    "ForegroundColor": 15,
    "Text": "1. Animals"
  },
  {
    "BackgroundColor": 0,
    "ForegroundColor": 15,
    "Text": "2. Create a new settings"
  },
  {
    "BackgroundColor": 15,
    "ForegroundColor": 0,
    "Text": "Please enter your choice: "
  },
  {
    "BackgroundColor": 15,
    "ForegroundColor": 0,
    "Text": "Sorry, out of order."
  },
  {
    "BackgroundColor": 15,
    "ForegroundColor": 0,
    "Text": "Goodbye."
  },
  {
    "BackgroundColor": 15,
    "ForegroundColor": 0,
    "Text": "Invalid choice. Try again."
```

7. Zastąpić obecnie istniejące definicje ekranów z wpisanym na zasadzie 'hardcoded' informacją pochodzącą z pliku JSON

Przykład wywołania metody PrintLine w klasie MainScreen. Jest ona wywoływana w ten sam sposób również na innych ekranach

```
public override void Show()
{
    while (true)
    {
        Console.ResetColor();
        _screenDefinition = ScreenDefinitionService.Load(ScreenDefinitionJson);
        ScreenDefinitionService.PrintLine(_screenDefinition.Screens, 0);
        ScreenDefinitionService.PrintLine(_screenDefinition.Screens, 1);
        ScreenDefinitionService.PrintLine(_screenDefinition.Screens, 2);
        ScreenDefinitionService.PrintLine(_screenDefinition.Screens, 3);
        ScreenDefinitionService.PrintLine(_screenDefinition.Screens, 4);
        ScreenDefinitionService.PrintLine(_screenDefinition.Screens, 5);
```

8. Dodać możliwość wyświetlenia historii wyborów poprzez zapamiętanie jej i wyświetlenie na górze ekranu. Czyli np Main Screen -> Mammals -> Dog

Tak wyglądają linie wyświetlające historię połączeń.

```
"Screens": [
  {
    "BackgroundColor": 13,
    "ForegroundColor": 15,
    "Text": "MainSceen->AnimalsScreen->MammalsScreen"
  },
```

```
"Screens": [
  {
    "BackgroundColor": 14,
    "ForegroundColor": 0,
    "Text": "MainSceen->AnimalsScreen->MammalsScreen->DogScreen"
  },
```

Tak wyglądają programy w działaniu

```
MainSceen->AnimalsScreen->MammalsScreen
Your available choices are:
0. Exit
1. Dogs
2. Elephants
3. Orangutans
4. Beavers
Please enter your choice:
```

```
MainSceen->AnimalsScreen->MammalsScreen->DogScreen
Your available choices are:
0. Exit
1. List all dogs
2. Create a new dog
3. Delete existing dog
4. Modify existing dog
Please enter your choice:
```