Programowanie Obiektowe

Kyrylo Horiunov
kirillgorunov294@gmail.com
144135

10.09.2023

https://github.com/Myrzei4/OOP/tree/final

Dodano:

- Projekt: SampleHierarchies.Services.Tests
- Klasy: ScreenDefinition, ScreenLineEntry, ScreenDefinitionService
- Jsons: AfricanElephantsScreenLines, AnimalsScreenLines, DogsScreenLines, GrizzlyBearsScreenLines, MainScreenLines, MammalsScreenLines, PolarBearsScreenLines, SettingsScreenLines

```
namespace TestProject1;

[TestClass]
0 references
public class UnitTest1
{
    [TestMethod]
    0 references
    public void Load_ValidJsonFile_ReturnsScreenDefinition()
    {
        // Arrange
        string jsonFileName = "valid.json"; // Specify the path to an existing JSON file
        ScreenDefinition screenDefinition = new ScreenDefinition();
        screenDefinition.LineEntries.Add(new ScreenLineEntry(ConsoleColor.White, ConsoleColor.Blue, "aboba"));
        string jsonContent = JsonConvert.SerializeObject(screenDefinition, Formatting.Indented);
        File.WriteAllText(jsonFileName, jsonContent); // Create a temporary JSON file with data
        // Act
        ScreenDefinition result = ScreenDefinitionService.Load(jsonFileName);

        // Assert
        Assert.IsNotNull(result);
        Assert.AreEqual(3, result.LineEntries.Count); // An example of checking that a ScreenDefinition object with one block is returned
    }

    [TestMethod]
    0 references
    public void Load_NonExistentJsonFile_ReturnsNull()
    {
        // Arrange
        string jsonFileName = "nonexistent.json"; // Specify a non-existent JSON file

        // Act
        ScreenDefinition result = ScreenDefinitionService.Load(jsonFileName);

        // Assert
        Assert.IsNull(result); // Expect the returned value to be null
    }

    [TestMethod]
    0 references
    public void Save_ValidScreenDefinition_SavesToFile()
    {
        // Arrange
        string jsonFileName = "valid.json"; // Specify the path to an existing JSON file
```

```
namespace SampleHierarchies.Data
{

    public class ScreenLineEntry
    {

        public ConsoleColor BackgroundColor { get; set; }

        public ConsoleColor ForegroundColor { get; set; }

        public string? Text { get; set; }


        public ScreenLineEntry(ConsoleColor backgroundColor, ConsoleColor foregroundColor, string text)
        {
            BackgroundColor = backgroundColor;
            ForegroundColor = foregroundColor;
            Text = text;
        }
    }
}
```

```
namespace SampleHierarchies.Data
{
    11 references
    public class ScreenDefinition
    {
        public List<ScreenLineEntry> LineEntries = new List<ScreenLineEntry>
        {
            new ScreenLineEntry(ConsoleColor.Green,
                ConsoleColor.White,
                "ScreenEnrty")
        };
    }
}
```

```
public static class ScreenDefinitionService
{
    /// <summary>
    /// Loads a screen definition from a JSON file.
    /// </summary>
    /// <param name="jsonFileName"></param>
    /// <returns></returns>

    public static ScreenDefinition? Load(string jsonFileName)
    {
        try
        {
            if (File.Exists(jsonFileName))
            {
                string jsonContent = File.ReadAllText(jsonFileName);
                return JsonConvert.DeserializeObject<ScreenDefinition>(jsonContent);

            }
            else
            {
                // File does not exist
                return null;
                throw new FileNotFoundException("JSON file {0} does not exist.", jsonFileName);
            }
        }
        catch
        {
            // Handle loading errors
            throw new Exception();
        }
    }
```

```
2 references | 2/2 passing
public static bool Save(ScreenDefinition screenDefinition, string jsonFileName)
{
    try
    {
        if (screenDefinition == null)
        {
            return false;
        }
        else
        {

            string jsonContent = JsonConvert.SerializeObject(screenDefinition, Formatting.Indented);
            File.WriteAllText(jsonFileName, jsonContent);
            return true;
        }
    }
    catch
    {
        // Saving error
        return false;
    }
}
```

```
"LineEntries": [
  {
    "BackgroundColor": 9,
    "ForegroundColor": 10,
    "Text": "Your available choices are: "
  },
  {
    "BackgroundColor": 9,
    "ForegroundColor": 10,
    "Text": "0. Exit"
  },
  {
    "BackgroundColor": 9,
    "ForegroundColor": 10,
    "Text": "1. List all dogs"
  },
  {
    "BackgroundColor": 9,
    "ForegroundColor": 10,
    "Text": "2. Create a new dog"
  },
  {
    "BackgroundColor": 9,
    "ForegroundColor": 10,
    "Text": "3. Delete existing dog" //5
  },
  {
    "BackgroundColor": 9,
    "ForegroundColor": 10,
    "Text": "4. Modify existing dog"
  },
  {
    "BackgroundColor": 9,
    "ForegroundColor": 10,
    "Text": "Please enter your choice: "
  },
  {
    "BackgroundColor": 9,
    "ForegroundColor": 10,
    "Text": "Going back to parent menu."
  },
  {
    "BackgroundColor": 9,
    "ForegroundColor": 12,
    "Text": "Invalid choice. Try again." ///9
```

Zmodyfikowano:

- Klasy: każdy screen: AfricanElephantsScreen, AnimalsScreen, DogsScreen, GrizzlyBearsScreen, MainScreen, MammalsScreen, PolarBearsScreen, Screen, SettingsScreen

```
/// <summary>
/// This part of code used for manipulatin with arrowkeys
/// </summary>
string? choiceAsString = "";
for (int i = 0; i <= 5;)
{
    if (i > 4)
    {
        i = 0;
    }
    if (i == -1)
    {
        i = 4;
    }
    ShowHistory(history);
    DisplayLine(1);
    for (int line = 0; line <= 4; line++)
    {
        if( line == -1)
        {
            line = 5;
        }
        if( line > 4)
        {
            line = 0;
        }
        if (line == i)
        {
            Console.BackgroundColor = ConsoleColor.DarkBlue;
            DisplayLineWOutColor(line + 2);
        }
        else
        {
            DisplayLine(line + 2);
        }
    }
    DisplayLine(7);
    Console.Write(i);
    ConsoleKeyInfo keyInfo = Console.ReadKey();
    if (keyInfo.Key == ConsoleKey.UpArrow)
    {
        Console.Clear();
        i--;
    }
    else if (keyInfo.Key == ConsoleKey.DownArrow)
    {
        Console.Clear();
        i++;
    }
    else if (keyInfo.Key == ConsoleKey.Enter)
    {
        Console.Clear();
        choiceAsString = i.ToString();
        PolarBearsScreenChoices choice = (PolarBearsScreenChoices)Int32.Parse(choiceAsString);
        break;
    }
```

```
            case MainScreenChoices.Exit:
                DisplayLine(6);
                history.RemoveAt(history.Count - 1);
                Console.ResetColor();
                return;

        }
    }
    catch
    {
        DisplayLine(7);
    }
```

```
public void DisplayLine(int lineNumber)
{
    try
    {
        if (ScreenDefinition != null)
        {
            var lineEntry = ScreenDefinition.LineEntries[lineNumber];
            Console.BackgroundColor = lineEntry.BackgroundColor;
            Console.ForegroundColor = lineEntry.ForegroundColor;
            Console.WriteLine();
            Console.Write(lineEntry.Text);
        }
    }
    catch (Exception ex)
    {
        Debug.WriteLine($"Error: {ex}");
    }
    finally
    {
        Console.ResetColor();
    }
}

/// <summary>
/// Method for displaying text line by line without color
/// </summary>
/// <param name="lineNumber"></param>
8 references
public void DisplayLineWOutColor(int lineNumber)
{
    try
    {
        if (ScreenDefinition != null)
        {
            var lineEntry = ScreenDefinition.LineEntries[lineNumber];
            Console.WriteLine();
            Console.Write(lineEntry.Text);
        }
    }
    catch (Exception ex)
    {
        Debug.WriteLine($"Error: {ex}");
    }
    finally
    {
        Console.ResetColor();
    }
}

8 references
public void ShowHistory(List<string> history)
{
    Console.ResetColor();
    foreach (var screen in history)
    {
        Console.Write($"=> {screen} ");
    }
    Console.WriteLine() ;
}
```

```
=> Main Screen => Animals => Mammals => African Elephants

Your available choices are:
0. Exit
1. List all african elephants
2. Create a new african elephant
3. Delete existing african elephant
4. Modify existing african elephant
Please enter your choice: 2
```