

## Programowanie Obiektowe: Zadanie #2

Imię i nazwisko: Bazyliak Stanislav

Adres email: darkplay084@gmail.com

Nr. albumu: 143067

Data: 25.05.2023

Git: <https://github.com/darkp111/SampleHierarchies.App.git>

## Rozdział 1 – Zadania do zrealizowania

Po wykonaniu zadań 1-10, dodałem klasy, które mogą być użyte do wypisania ich linia po linii z pliku json, zmieniając również ForegroundColor i BackgroundColor.

Klasy, które dodałem i wykorzystałem:

❓ Klasa ScreenLineEntry:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace SampleHierarchies.Data.ScreenSettings
{
    9 references
    public class ScreenLineEntry
    {
        7 references
        public string? BackgroundColor { get; set; }
        7 references
        public string? ForegroundColor { get; set; }
        6 references
        public string? Text { get; set; }
    }
}
```

❓ Klasa ScreenDefinition:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace SampleHierarchies.Data.ScreenSettings
{
    25 references
    public class ScreenDefinition
    {
        12 references
        public List<ScreenLineEntry>? LineEntries { get; set; }
    }
}
```

❓ Dodać klasę statyczną o nazwie ScreenDefinionService , która zawiera następujące metody:

1. ScreenDefinition Load(string jsonFileName)
2. bool Save(ScreenDefinition screenDefinition, string jsonFileName)

```

public class ScreenDefinitionService : IScreenDefinitionService
{
    7 references
    public ScreenDefinition Load(string jsonFileName)
    {
        string jsonContent = File.ReadAllText(jsonFileName);
        ScreenDefinition screenDefinition;

        try
        {
            screenDefinition = JsonConvert.DeserializeObject<ScreenDefinition>(jsonContent);
        }
        catch (JsonException ex)
        {
            throw new Exception("Error deserializing JSON: " + ex.Message);
        }

        if (screenDefinition == null)
            throw new Exception("Deserialization resulted in null object.");

        return screenDefinition;
    }

    2 references
    public bool Save(ScreenDefinition screenDefinition, string jsonFileName)
    {
        try
        {
            string jsonContent = JsonConvert.SerializeObject(screenDefinition);
            File.WriteAllText(jsonFileName, jsonContent);
            return true;
        }
        catch (Exception ex)
        {
            Console.WriteLine("Error saving screen definition: " + ex.Message);
            return false;
        }
    }
}

```

```

99+ references
public void PrintScreen(ScreenDefinition content, int lineNumber, string jsonPath)
{
    content = Load(jsonPath);
    if (content == null)
    {
        Console.WriteLine("Invalid content download");
    }
    else
    {
        if (content.LineEntries[lineNumber].BackgroundColor == null & content.LineEntries[lineNumber].ForegroundColor == null)
        {
            Console.WriteLine("Content not found.");
        }
        else
        {
            Console.BackgroundColor = (ConsoleColor)Enum.Parse(typeof(ConsoleColor), content.LineEntries[lineNumber].BackgroundColor, true);
            Console.ForegroundColor = (ConsoleColor)Enum.Parse(typeof(ConsoleColor), content.LineEntries[lineNumber].ForegroundColor, true);
            Console.WriteLine(content.LineEntries[lineNumber].Text);
            Console.ResetColor();
        }
    }
}

```

class System.String  
Represents text as a sequence of UTF-16 code units.

❖ Dodać interfejs do wspomniane w punkcie 4 klasy o nazwie IScreenDefinitionService.

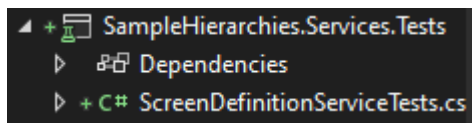
```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace SampleHierarchies.Data.ScreenSettings
{
    16 references
    public interface IScreenDefinitionService
    {
        7 references
        ScreenDefinition Load(string jsonFileName);
        2 references
        bool Save(ScreenDefinition screenDefinition, string jsonFileName);
        99+ references
        public void PrintScreen(ScreenDefinition content, int lineNumber, string jsonPath);
        8 references
        public string GetText(ScreenDefinition content, int lineNumber, string jsonPath);
        13 references
        public ConsoleColor GetForegroundColor(ScreenDefinition content, int lineNumber, string jsonPath);
        13 references
        public ConsoleColor GetBackgroundColor(ScreenDefinition content, int lineNumber, string jsonPath);
        8 references
        public string[] SplitStringByNewLine(string input);
    }
}

```

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



```
namespace SampleHierarchies.Tests
{
    [TestClass]
    public class ScreenDefinitionServiceTests
    {
        private const string TestJsonFileName = "test.json";

        [TestMethod]
        public void Load_ValidJsonFile_ReturnsScreenDefinition()
        {
            // Arrange
            var service = new ScreenDefinitionService();
            var screenDefinition = new ScreenDefinition
            {
                LineEntries = new List<ScreenLineEntry>
                {
                    new ScreenLineEntry { BackgroundColor = "Black", ForegroundColor = "White", Text = "Sample Text" }
                }
            };
            string jsonContent = JsonConvert.SerializeObject(screenDefinition);
            File.WriteAllText(TestJsonFileName, jsonContent);

            try
            {
                // Act
                var result = service.Load(TestJsonFileName);

                // Assert
                Assert.IsNotNull(result);
                // Add more assertions to verify the loaded screenDefinition object
            }
            finally
            {
                // Cleanup
                File.Delete(TestJsonFileName);
            }
        }
    }
}
```

```
[TestMethod]
[ExpectedException(typeof(Exception))]
public void Load_InvalidJsonFile_ThrowsException()
{
    // Arrange
    var service = new ScreenDefinitionService();
    File.WriteAllText(TestJsonFileName, "invalid json content");

    try
    {
        // Act
        service.Load(TestJsonFileName);
    }
    finally
    {
        // Cleanup
        File.Delete(TestJsonFileName);
    }
}

[TestMethod]
public void Save_ValidScreenDefinition_ReturnsTrue()
{
    // Arrange
    var service = new ScreenDefinitionService();
    var screenDefinition = new ScreenDefinition
    {
        LineEntries = new List<ScreenLineEntry>
        {
            new ScreenLineEntry { BackgroundColor = "Black", ForegroundColor = "White", Text = "Sample Text" }
        }
    };
};
```

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

```

1  using SampleHierarchies.Data.ScreenSettings;
2  using SampleHierarchies.Services;
3
4  namespace SampleHierarchies.Gui;
5
6  /// <summary>
7  /// Abstract base class for a screen.
8  /// </summary>
9  public abstract class Screen
10 {
11     #region Public Methods
12
13     /// <summary>
14     /// Show the screen.
15     /// </summary>
16     ///
17     public SettingsService? _settingsService;
18
19     public string? _screenDefinitionJson;
20
21     public virtual void Show()
22     {
23         Console.WriteLine("Showing screen");
24     }
25
26     #endregion // Public Methods
27 }
28

```

- ❖ Nadpisać tę wartość w każdej klasie potomnej.










```

0 references
public MainScreen(
    IDataService dataService,
    AnimalsScreen animalsScreen,
    ISettings settings,
    IScreenDefinitionService screenDefinitionService,
    MenuManager menuManager)
{
    _dataService = dataService;
    _animalsScreen = animalsScreen;
    _settings = settings;
    _screenDefinitionService = screenDefinitionService;
    _screenDefinitionJson = "MainMenu.json";
    this.menuManager = menuManager;
}

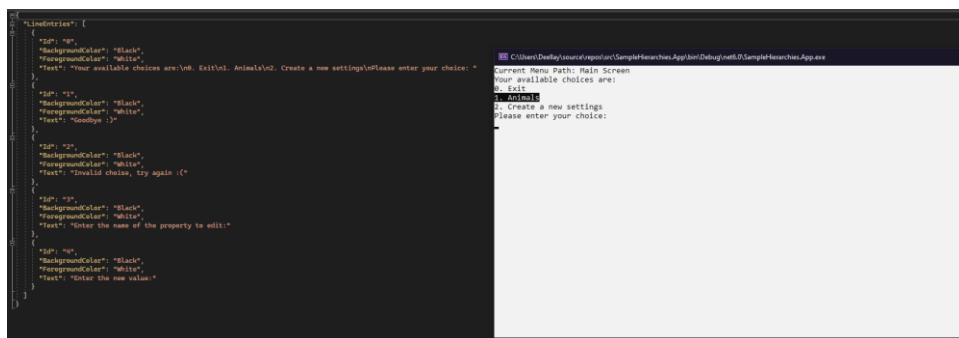
```

Wszystkie inne klasy również mają tę wartość

- ❖ Dodać pliki JSON dla każdego z ekranów z definicją ich zawartości.

 AnimalsMenu.json	15.07.2023 18:14	JSON File	2 KB
 Animalss.json	17.05.2023 13:00	JSON File	1 KB
 BottlenoseWhalesMenu.json	15.07.2023 18:50	JSON File	4 KB
 DogsMenu.json	15.07.2023 18:31	JSON File	3 KB
 LionsMenu.json	15.07.2023 18:57	JSON File	4 KB
 MainMenu.json	15.07.2023 17:56	JSON File	1 KB
 mamals.json	17.05.2023 11:47	JSON File	1 KB
 MammalsMenu.json	15.07.2023 18:10	JSON File	1 KB
 menu.json	15.07.2023 14:26	JSON File	15 KB

- ❖ Wstawić renderowanie ekranu na podstawie kolorystyki wskazanej w plik JSON.



- ❖ Dodać możliwość poruszania się po ekranie za pomocą klawiszy strzałek. Po wybraniu wiersza i naciśnięciu Enter powinno sterowanie być przekazane do wybranego ekranu.

```
C:\Users\Deellay\source\repos\src\SampleHierarchies.App\bin\Debug\net6.0\SampleHierarchies.App.exe
Current Menu Path: Main Screen -> Animals Screen -> Mammals Screen
Your available choices are:
0. Exit
1. Dogs
2. Polar Bears
3. Lions
4. Bottlenose Whales
Please enter your choice:

switch (keyInfo.Key)
{
    case ConsoleKey.UpArrow:
        selectedIndex = (selectedIndex - 1 + menuItems.Length) % menuItems.Length;
        Thread.Sleep(150);
        break;

    case ConsoleKey.DownArrow:
        selectedIndex = (selectedIndex + 1) % menuItems.Length;
        Thread.Sleep(150);
        break;

    case ConsoleKey.Enter:
        if (selectedIndex == menuItems.Length - 1)
        {
            return; // Вихід з програми
        }
        else
        {
            Console.WriteLine("\nYou choose: " + menuItems[selectedIndex]);
            try
            {
                MainScreenChoices choice = (MainScreenChoices)selectedIndex - 1;
                switch (choice)
                {
                    case MainScreenChoices.Animals:
                        Console.Clear();
                        _animalsScreen.Show();
                        break;

                    case MainScreenChoices.Settings:
                        EditJsonFile("settings.json");
                        break;

                    case MainScreenChoices.Exit:
                        _screenDefinitionService.PrintScreen(_currentScreenDefinition, 1, _screenDefinitionJson);
                        return;
                }
            }
            catch { }
        }
    }
}
```

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

```
C:\Users\Deellay\source\repos\src\SampleHierarchies.App\bin\Debug\net6.0\SampleHierarchies.App.exe
Current Menu Path: Main Screen -> Animals Screen -> Mammals Screen -> Dogs Screen
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:
```

Zaimplementowałem to tworząc klasę MenuManager:

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6
7 namespace SampleHierarchies.Gui
8 {
9     16 references
10     public class MenuManager
11     {
12         private List<string> menuPath = new List<string>();
13
14         7 references
15         public void AddToMenuPath(string menuName)
16         {
17             if (!menuPath.Contains(menuName))
18             {
19                 menuPath.Add(menuName);
20             }
21         }
22
23         6 references
24         public void RemoveLastFromMenuPath()
25         {
26             if (menuPath.Count > 0)
27             {
28                 menuPath.RemoveAt(menuPath.Count - 1);
29             }
30         }
31
32         20 references
33         public string GetCurrentMenuPath()
34         {
35             return string.Join(" -> ", menuPath);
36         }
37     }
38 }
```