|  |
| --- |
| Merito University |
| Programowanie Obiektowe |
| Ćwiczenia 2 - zadania |

|  |
| --- |
| Autor: Lesław Pawlaczyk  2023/04/20 |

Spis treści

[Rozdział 1 – Zadania do zrealizowania 2](#_Toc132844718)

[Rozdział 2 – Ocenianie 3](#_Toc132844719)

[Bibliografia 4](#_Toc132844720)

# Rozdział 1 – Zadania do zrealizowania

**Zadania podstawowe**:

1. Dodać klasę ScreenLineEntry o następujących polach:

* BackgroundColor
* ForegroundColor
* Text

Klasa ta będzie definiować pojedynczą linię na ekranie.

1. Dodać klasę o nazwie ScreenDefinition o następujących polach:

* LineEntries - typu List<ScreenLineEntry>

1. Klasy z punktów 1 oraz rozszerzyć o definicje.
2. Dodać klasę statyczną o nazwie ScreenDefinionService , która zawiera następujące metody

* ScreenDefinition Load(string jsonFileName)
* bool Save(ScreenDefinition screenDefinition, string jsonFileName)

Referencja do tej klasy powinna być przekazana w taki sam sposób jak jest przekazywana referencja do SettingsService do klasy bazowej Screen.cs do konstruktora poprzez wstrzykiwanie zależności.

1. Dodać interfejs do wspomniane w punkcie 4 klasy o naziwa IScreenDefinitionService.
2. Dodać nowy projekt o nazwie SampleHierarchies.Services.Tests, do którego należy dodać unit testy klasy ScreenDefinitionService.cs
3. Dodać do klasy Screen.cs pole o nazwie ScreenDefinitionJson, które będzie przechowywało nazwę pliku z definicją ekranu.
4. Nadpisać tą wartość w każdej klasie potomnej.
5. Dodać pliki JSON dla każdego z ekranów z definicją ich zawartości.
6. Zastąpić obecnie istniejące definicje ekranów z wpisanym na zasadzie 'hardcoded' informacją pochodzącą z pliku JSON.

**Ekrany zrobione Ćwiczeniu 1 powinny wszystkie mieć zaimplementowane wskazane zmiany**. Należy unikać jak to możliwe powtarzania kodu.

**Zadania dodatkowe**:

1. Wstawić renderowanie ekranu na podstawie kolorystyki wskazanej w plik JSON.
2. Dodać możliwość poruszania się po ekranie za pomocą klawiszy strzałek. Po wybraniu wiersza i naciśnięciu Enter powinno sterowanie być przekazane do wybranego ekranu.
3. Dodać możliwość wyświetlenia historii wyborów poprzez zapamiętanie jej i wyświetlenie na górze ekranu. Czyli np Main Screen -> Mammals -> Dog
4. Dodać pole o nazwie ListEntriesPl do klasy ScreenDefinition. Uzupełnić pliki JSON o wersję polską interfejsu użytkownika. Dodać do settings opcję wyboru języka. Opcja wyboru powinna być zapisywana za pomocą SettingsService i dodana do settings.json. Wybór powinien od razu zmienić wygląd kolejnych ekranów na wybrany język. Dodać odpowiednie wersje językowe wewnątrz plików definicji ekranów.

# Rozdział 2 – Ocenianie

**Rozwiązania powinny być indywidualne**.

1. Wykonać zadania od 1 do 10.
2. Wynik w postaci linku do repozytorium na GitHub oraz sprawozdanie w PDF. Sprawozdanie w PDF powinno się nazywać: ProgramowanieObiektowe\_Zadanie2\_20230420\_Rozwiazanie\_ImieNazwisko\_XXXXX.pdf, gdzie ImieNazwisko zastąpić swoim imieniem i nazwiskiem a XXXXX numerem albumu.
3. Repozytorium na Github nie powinno zawierać katalogów obj oraz bin. W katalogu głównym dodać sprawozdanie.

**Strona tytułowa**:

1. Imię i nazwisko.
2. Adres email.
3. Numer albumu.
4. Data.
5. Link do repo na Github.

**Sposób oceniania**:

1. Ocena 3.0 – zadania nr 1 do 10
2. Ocena 3.5 – zadania nr 1 do 11.
3. Ocena 4.0 – zadania nr 1 do 12.
4. Ocena 4.5 – zadania nr 1 do 13
5. Ocena 5.0 – zadania nr 1 do 14.

**Oceniane będą**:

1. Jakość kodu.
2. Komentarze.
3. Poprawność działania programu.

**Czas do wykonania**:

1. Do 2023/09/15.

**Uwaga:**

Przy zauważeniu, że kod powtarza się z kodem innego studenta rozwiązanie zostanie uznane za plagiat i ocenione na ocenę „ndst”.

# Bibliografia

1. <https://learn.microsoft.com/pl-pl/dotnet/fundamentals/>
2. <https://learn.microsoft.com/pl-pl/dotnet/csharp/fundamentals/tutorials/oop>
3. <https://learn.microsoft.com/pl-pl/dotnet/standard/design-guidelines/common-design-patterns>
4. <https://learn.microsoft.com/pl-pl/visualstudio/get-started/csharp/tutorial-console?view=vs-2022>