LearnIt - dokumentacja

Programowanie obiektowe 2017

Maciej Draguła

czerwiec 2017

1 Alfabetyczny spis klas

- Base
- Exercise1
- Exercise2
- Exercise3
- Exercise4
- Frame
- Intro
- Login
- MainProgram
- NewUserWindow
- StartWindow
- UserMenu
- WhatToLearn

2 Opis projektu oraz poszczególnych klas

2.1 Ogólny opis projektu

LearnIt jest niewielką aplikacją komputerową służącą do nauki słówek i haseł za pomocą tzw. fiszek. Program został stworzony jako projekt końcowy na przedmiocie *Programowanie Obiektowe* na Uniwersytecie Wrocławskim w semestrze letnim roku akademickiego 2016/2017 przez Agnieszkę Dudek, Julię Majkowską oraz Macieja Dragułę. Jest on napisany w języku Java z wykorzystaniem biblioteki graficznej Swing.

Po uruchomieniu programu widoczne jest okno logowania. Użytkownik powinien założyć konto klikając przycisk *New user*. Po wybraniu nazwy użytkownika i hasła należy zatwierdzić stworzenie użytkownika przyciskiem *Create new account*. Po zapoznaniu się z ogólnymi wskazówkami należy wybrać, czy użytkownik chce załadować istniejącą bazę z pliku, czy też stworzyć własną.

Podczas tworzenia własnej bazy mamy możliwość, oprócz obowiązkowych pól takich jak: *Category*, *Phrase*, *Meaning*, dodać własne. Po zapisaniu nowostworzonej bazy należy zapsiać ją do pliku i załadować. Następnie otwiera się nowe menu, w którym możemy wybrać, czy chcemy dodać nowe fiszki do załadowanej bazy (przycisk *Add*), uczyć się nowych albo powtarzać wcześniej przyswojone.

W trybie *Learn* użytkownik winien wybrać kategorię oraz liczbę fiszek, których chce się uczyć (dostępna jest opcja *wszytskie*). Tryb *Learn* jest podzielony na cztery etapy. W pierwszym program wyświetla hasła oraz ich znaczenia, należy wtedy próbować zapamiętać frazy. Użytkownik sam decydyje, kiedy chce uczyć się kolejnego słówka klikając przycisk *Continue*. Następnie rozpoczyna się pierwsze ćwiczenie, w którym zostaje wyświetlone pytanie, a po kliknięciu *Show answer* odpowiedź. Użytkownik sam decyduje, czy zapamiętał hasło czy nie klikając jeden z przycisków: *Fail* lub *OK*. Potem następuje druga faza nauki, w której wyświetlona jest część odpowiedzi i należy uzupełnić ją brakującymi literkami. W przypadku błędnego wprowadzenia tekstu należy kliknąć *Reset answer*. Program weryfikuje poprawność odpowiedzi i w przypadku błędu wyświetli poprawne rozwiązanie. W czwartej fazie nauki użytkownik otrzymuje ćwiczenie, w którym, bez żadnych podpowiedzi, powinien wprowadzić odpowiedź na zadane pytanie lub przetłumaczyć frazę. Podobnie jak w fazie trzeciej w przypadku błędnej odpowiedzi program wyświetli użytkownikowi prawidłową odpowiedź. Jeżeli na któreś z ćwiczeń została udzielona niepoprawna odpowiedź, program doda fiszkę na koniec kolejki pytań i będzie to robił tak długo, aż użytkownik nie udzieli bezbłędnych odpowiedzi.

Tryb *Repeat* służy do powtarzania haseł poznanych w trybie *Learn*. Program, w zależności od poprawności odpowiedzi użytkownika, sam decyduje w jakim odstępie czasowym dane słówko należy powtórzyć. Odstęp jest wydłużany przy kolejnych poprawnych odpowiedziach oraz skracaniu przy błędnych.

2.2 Opis mojej części pracy

Jak każdy członek zespołu brałem udział w projektowaniu całego programu, natomiast moim głównym zadaniem była implementacja interfejsu graficznego z wykorzystaniem biblioteki Swing. Prawie wszytskie stworzone przeze mnie klasy są rozszerzeniem klasy *JPanel* lub *JFrame*. Obiekty stwarzane przeze mnie w programie są w większości kontenerami, które wyświetlają

menu albo fiszki podawane przez algorytm Agnieszki Dudek. Niektóre klasy bezpośrendnio implementują interfejsy *ActionListener* oraz *ChangeListener*, do innych natomiast jest przekazywana referencja do obiektu klasy *Frame*, którego głównym zadaniem jest odbieraniem sygnałów od kolejnych przysisków.
Każda stworzona klasa rozszerzająca *JPanel* korzysta z managera layoutów o nazwie *GridBagLayout*. Za pomocą obiektu klasy *GridBagConstraints* defniowałem położenie przycisków, etykiet oraz pól do wprowadzania tekstu, jak również rozmiar wcięć i dopasowanie pól do szerowości ramki. Implemetacja interfejsu graficznego we wszytskich klasach sprowadza się do implementacji konstruktów oraz ewentualnie metody *ActionPerformed* wymaganej przez interfejsy *ActionListener* oraz *ChangeListener*.

2.3 Klasa: Base

extends Jpanel

Zadaniem klasy Base jest wyświetlenie dwóch przycisków, które pozwalają stworzyć nową bazę fiszek albo załadować z pliku. Informacja o sygnale pochodzącym od przycisków *Create new base* oraz *Load from file* jest przekazywana do obiektu klasy *Frame*.

2.4 Klasa: Exercise1

extends Jpanel

Klasa *Exercise1* to interfejs ćwiczenia, w którym użytkownić musi samodzielnie wprowadzić odpowiedź z klawiatury. Argumentami konstruktora są referencje do aktualnie używanej ramki oraz do pytania. Panel składa się a dwóch podpanelów (panel pytania i panel odpowiedzi) oraz przycisku *Continue*, z którego sygnał jest przekazywany do ramki. W panelu pytania umieszczona jest etykieta z pytaniem na białym tle, natomiast w panelu odpowiedzi znajduje się pole tekstowe do wprowadzania odpowiedzi.

2.5 Klasa: Exercise2

extends JPanel implements ActionListener

Klasa *Exercise2* to interfejs ćwiczenia, którym użytkownik odpowiada samodzielnie na zadane pytanie bądź tłumaczy frazę, a następnie wybiera, czy udzielił prawidłowej odpowiedzi. W konstruktorze tworzone są (podobnie jak w ćwiczeniu nr 1) panel pytania i odpowiedzi. W panelu z pytaniem

umieszczona jest etykieta z pytaniem na białym tle. Panel pytania początkowo zawiera pustą etykietę z odpowiedzią. Poniżej panelów znajdują się przyciski *Show answer*, *Fail*, *OK*, przy czym dwa ostatnie są początkowo ukryte.
Klasa implementuje interfejs *ActionListener*, zatem jest również zaimplementowana metoda *actionPerformed*. Przycisk *Show answer* jest połączony z lokalnym Action Listenerem i po kilknięciu tego przycisku jest on ukrywany, natomiast pojawiają się przyciski *Fail* oraz *OK* połączone z ramką typu *Frame*. Wtedy wyświetlana jest również odpowiedź.

2.6 Klasa: Exercise3

extends JPanel implements ActionListener

Klasa *Exercise3* to interfejs graficzny ćwiczenia, w którym użytkownik wprowadza tylko konkretne litery do odpowiedzi. Konstruktor pobiera referencję do ramki, pytanię, odpowiedź (z podstawionym znakiem podkreśleń zamiast niektórych liter) oraz tablicę liter, które użytkownik powinien wprowadzić. W panelu zostaje umieszczone pole z pytaniem, pole z odpowiedzią, przyciski z literami, przycisk *Reset answer* oraz przycisk *Continue*.
Klasa implementuje metodę *actionPerformed*, której zadaniem jest odbieranie sygnałów od odpowiednich przycisków-liter oraz podstawianie tych liter do odpowiedzi. Kiedy przychodzący sygnał pochodzi od przycisku *Reset answer*, to metoda zamienia aktualną etykietę odpowiedzi, na startową z początkową liczbą znaków podkreślenia.

2.7 Klasa: Exercise4

extends Jpanel

Klasa *Exercise4* to interfejs ćwiczenia, w którym użytkownik widzi pytanie i odpowiedź i stara się ją zapamiętać. Panel złożony jest z dwóch podpanelów: panelu pytania i odpowiedzi, oraz przycisku *Next*. Konstruktor tworzy wszytskie wyżej wymienione pola oraz łączy przycisk z aktualną ramką, do której referencja jest podana jako argument konstruktora.

2.8 Klasa: Frame

extends JFrame implements ActionListener

Klasa *Frame* jest najważniejszą klasą w całym programie, w której odbywa się wyświetlanie większość paneli przygotowanych w innych klasach. Konstruk-

tor umieszcza w panelu ramki obiekt klasy *Intro*, pobierając jednocześnie zalogowanego użytkownika.
Metoda *actionPerformed* dopasowuje źródło sygnału do różnych przycisków, które mogą zostać wyświetlone w ramce oraz które zmieniają aktualny konetner (panel) w ramce. Ponieważ następuje próba rzutowania źródła sygnału na różne podklasy *JPanel* to metoda jest wydzielona strukturą *try catch*. Jeżeli sygnał pochodzi od danego przycisku to wykonuje następujące działanie:

- *btContinue* z klasy *Intro*: usuwa aktualny panel i dodaje panel klasy *Base*
- *btLogOut* z klasy *UserMenu*: zamyka aktualny ramkę typu *Frame* i otwiera okno startowe
- *btLearn* z klasy *UserMenu*: usuwa aktualny panel, pobiera mapę dostępnych kategorii oraz ilość kart w nich się znajdujących, listę kategorii, tworzy nowy panel klasy *WhatToLearn* i dodaje go do ramki
- *btContinue* z klasy *WhatToLearn*: usuwa aktualny panel, pobiera kolejkę fiszek na podstawie danych wprowadzonych przez użytkownika oraz dodaje do ramki nowy panel, którym jest odpowiednie ćwiczenie z fiszką z góry kolejki, bądź wraca do menu wyboru trybu
- *btContinue* z klasy *Exercise1*: usuwa aktualny panel, porównuje odpowiedź z poprawną odpowiedzią i albo umieszcza ćwiczenie na końcu kolejki i wyświetla fiszkę w formie ćwiczenia 4, albo wyświetla kolejne ćwiczenie z kolejki albo wraca do menu wyboru trybów
- *btOK* z klasy *Exercise2*: usuwa aktualny panel i dodaje nowy panel z ćwiczeniem i fiszką z góry kolejki lub wraca do menu wyboru trybów
- *btFail* z klasy *Exercise2*: usuwa aktualny panel, dodaje ćwiczenie na koniec kolejki i dodaje nowy panel z ćwiczeniem i fiszką z góry kolejki lub wraca do menu wyboru trybów
- *btContinue* z klasy *Exercise3*: usuwa aktualny panel, porównuje odpowiedź z poprawną odpowiedzią i albo umieszcza ćwiczenie na końcu kolejki i wyświetla fiszkę w formie ćwiczenia 4, albo wyświetla kolejne ćwiczenie z kolejki albo wraca do menu wyboru trybów
- *btLBase* z klasy *Base*: usuwa aktualny panel, dodaje nowy panel, którym jest obiekt klasy *UserMenu* oraz otwiera okno, za pomocą którego można załadować bazę z pliku

- *btCrNewBase* z klasy *Base*: otwiera okno pozwalające stworzyć własną bazę fiszek
- *btAdd* z klasy *UserMenu*: uruchamia okno pozwalające modyfikować aktualnie załadowaną bazę
- *btContinue* z klasy *Exercise4*: usuwa aktualny panel i dodaje nowy panel z ćwiczeniem i fiszką z góry kolejki lub wraca do menu wyboru trybów
- *btRepeat* z klasy *UserMenu*: usuwa aktualny panel, pobiera kolejkę fiszek przygotwanych do powtórki na dany dzień oraz dodaje do ramki nowy panel, którym jest odpowiednie ćwiczenie z fiszką z góry kolejki, bądź wraca do menu wyboru trybu

2.9 Klasa: Intro

extends JPanel implements ActionListener

Obiekt tej klasy jest panelem, na którym znajduje się tekst informujący o działaniu programu oraz przycisk *Continue*. Argumentem konstruktora jest ramka typu *Frame*, do której przekazywana jest infromacja o sygnale pochodzącej z przycisku.

2.10 Klasa: MainProgram

Jest to jedyna klasa nieimplementująca żadnego interfejsu ani nie rozszerzeająca żadnej innej klasy. Zawiera jedynie publiczną, statyczną metodę klasy void o nazwie *Main*, której zadaniem jest stworzenie obiektu okna startowego i uruchomienie interfejsu graficznego służącego do interakcji z użytkownikiem.

2.11 Klasa: NewUserWindow

extends JPanel implements ActionListener

Konstruktor tej klasy stwarza okno, w którym użytkownik stwarza nowy profil do nauki w programie. Okno zawiera jedno pole tekstowe dla nazwy użytkownika, dwa pola haseł oraz przycisk *Create new user*. Klasa impelmentuje metodę *actionPerformed*, która obiera sygnał od przycisku w oknie i wywołuje funckję, która tworzy (bądź nie) nowego użytkownika w bazie danych. Jeśli użytkownik wprowadzi niejednakowe hasła to pokaże się ostrze-

żenie w formie etykiety. Jeśli użytkownik zostanie stworzony poprawnie, to okno zostanie automatyczne zamknięte.

2.12 Klasa: StartWindow

extends JPanel implements ActionListener

Klasa *StartWindow* jest oknem programu, które pojawia się jako pierwsze po uruchomieniu. Na jednym panelu znajdują się pola do logowania się oraz etykiety opisujące pola tekstowe. Ponadto w oknie znajduje się przycisk, stwarza nowy obiekt klasy *NewUserWindow* pozwalający na stworzenie użytkownika. Metoda *actionPerformed* odbiera sygały od przycisków *Log in* oraz *New user*. W przypadku wybrania przycisku *Log in* za pomocą funkcji napisanej przez Agnieszkę Dudek następuje próba logowania danymi z pól tekstowych. Jeżeli logowanie wykonało się poprawnie, to okno zostaje zamknięte i zostaje utworzony nowy obiekt klasy *Frame* z argumentem *user*, który przechowuje dane zalogowanego użytkownika.

2.13 Klasa: UserMenu

extends JPanel

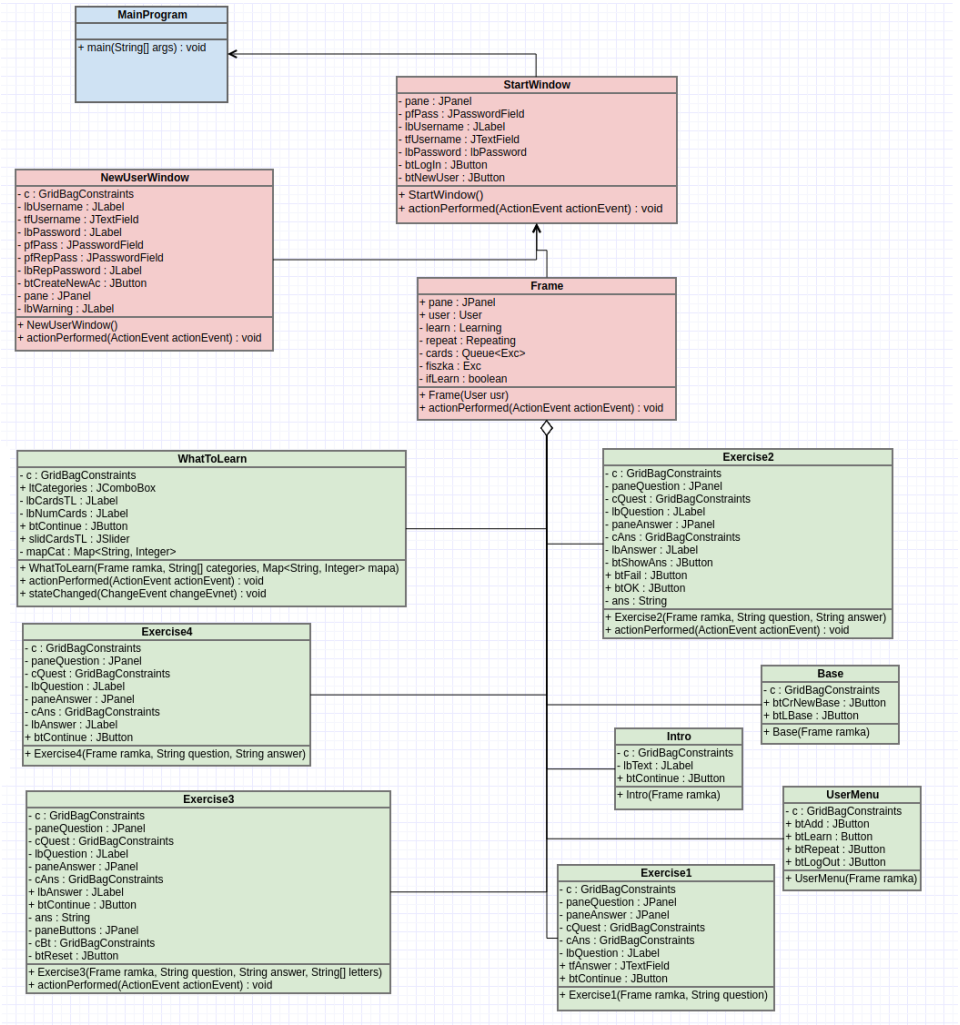
Klasa zawiera tylko jedną metodę - konstruktor pobierający ramkę typu *Frame*. Konstruktor ma za zadanie stworzenie menu wyboru trybów: *Add*, *Learn*, *Repeat*. W panelu znajduje się tak że przycisk *Log out* pozwalający się wylogować z aktualnego konta.

2.14 Klasa: WhatToLearn

extends JPanel implements ActionListener, ChangeListener

Klasa *WhatToLearn* to menu wyboru kategorii oraz liczby kart do nauki. Konstruktor pobiera referencję do ramki, tablicę kateogrii, oraz mapę (z nazwy kategorii w liczbę dostepnych kart). Na panelu umieszczony jest *JComboBox*, dzięki któremu można wybrać kategorię oraz suwak *JSlider*, którym można wybierać liczbę kart do nauki. Wybór użytkownik zatwierdza przycskiem *Continue* umieszczonym poniżej. Klasa implementuje metody *actionPerformed* oraz *stateChanged*, aby po wybraniu kategorii mieć do wyboru określoną liczbę kart oraz aby etykieta *lbNumCards* na bieżąco wyświetlała wybaną liczbę fiszek.

3 Diagram klas



4 Wzorzec projektowy

W mojej części projektu LearnIt można zidentyfikować wzorzec *Fasada*. Napisany przez mnie interfejs towrzy właśnie fasadę, za pomocą której użytkownik może komunikować się z bazą danych oraz oraz kolejką przechowującą kolejne fiszki. Nie występuje tutaj dziedziczenie pomiędzy stworzonymi przeze mnie klasami (rozszerzają one tylko gotowe *JFrame* i *JPanel* z biblioteki Swing), natomiast wykorzystują funkcje stworzone przez Julię Majkowską i

Agnieszkę Dudek.

5 Inne zastosowania

Klasy odpowiadające za logowanie się do systemu można wykrzystać do logowania się na konta w dowolnym projekcie. Wystarczy tylko podać odpowiednią bazę danych oraz okno, które ma zostać wyświetlone po udanym zalogowaniu.
Ponadto można zauważyć, że projekt jest przystowsowany do rozszerzeń. W łatwy sposób można dodać nowy rodzaj ćwiczenia oraz dodać go zestawu ćwiczeń w trybie *Learn*.