

JAVA : lab05 - Swing

implementacja aplikacji okienkowej przy pomocy biblioteki Swing języka Java

Materiały

1. [Layouty w Swingu 1](<https://docs.oracle.com/javase/tutorial/uiswing/layout/visual.html>)
2. [Layouty w Swingu 2](<https://examples.javacodegeeks.com/desktop-java/swing/java-swing-layout-example/>)
3. [Lista eventów w Swingu](<https://docs.oracle.com/javase/7/docs/api/javax/swing/event/package-summary.html>)
4. [Przykład obsługi eventów w Swingu 1](https://www.tutorialspoint.com/swing/swing_event_handling.htm)
5. [Przykład obsługi eventów w Swingu 2](<https://stackoverflow.com/questions/19122514/handling-events-in-java-swing>)
6. Jak można stworzyć generyczny komponent na bazie AbstractTableModel:
7. [Jak korzystać z AbstractTableModel?](<https://stackoverflow.com/questions/9845800/abstracttablemodel-tutorial>)
8. [Wykorzystanie refleksji do pobrania properties](<https://stackoverflow.com/questions/8524011/java-reflection-how-can-i-get-the-all-getter-methods-of-a-java-class-and-invoke>)

Zadania

1. Zaimplementuj lub wykorzystaj model z **lab02**, który dysponuje funkcjonalnościami do zarządzania studentami/uczniemi (np. dodaje/usuwa, zmienia stan studenta, ilość punktów itd. – jak w lab02).
2. **Zaprojektuj interfejs graficzny** do zarządzania dziennikiem, który obsługuje wszystkie metody z podpunktu 1.
3. **Zaimplementuj obsługę interfejsu graficznego** według następujących reguł:
 - 1. Interfejs składa się z dwóch list: studentów oraz grup. Dodatkowo można wyświetlać krótkie informacje na temat studentów
 - 2. Stwórz generyczny komponent na bazie
 - [JTable`](<https://docs.oracle.com/javase/7/docs/api/javax/swing/JTable.html>)Oraz
 - [AbstractTableModel`](<https://docs.oracle.com/javase/7/docs/api/javax/swing/table/AbstractTableModel.html>), który można wykorzystać zarówno dla studentów jak i dla grup. Więcej informacji nt. AbstractTableModel na [StackOverflow](<https://stackoverflow.com/questions/9845800/abstracttablemodel-tutorial>).
 - 3. Po wybraniu grupy (zaznaczeniu go na liście) wyświetla się dostępna w nim lista studentów.
 - 4. Zaznaczony obiekt ma zostać usunięty po naciśnięciu odpowiedniego guzika (remove),
 - 4.b Zaznaczony obiekt może zostać edytowany po naciśnięciu odpowiedniego guzika (change info)

- 5. Dodaj studenta lub dodaj grupę po naciśnięciu odpowiedniego przycisku, wprowadzając odpowiednie parametry za pomocą komponentu [JOptionPane](<https://docs.oracle.com/javase/7/docs/api/javafx/swing/JOptionPane.html>). Alternatywnie, przycisk może dodać wiersz do tabeli a wprowadzenie wartości odbywa się poprzez edycję w tabeli.
 - 6. Po naciśnięciu przycisku sort ma zostać wykonane sortowanie obiektów (np. względem maksymalnego obciążenia grupy, nazwiska, ilości punktów). ****Uwaga****: przekazuj referencje w odpowiedni sposób, wówczas taka operacja to wywołanie jednej metody.
 - 7. Pole _filter textbox_ służy do wprowadzania imienia studenta. Po naciśnięciu klawisza enter, w tabeli mają zostać wyświetlone osoby zgodne z wprowadzonym tekstem.
 - **[**Nieobowiązkowe**]**
Pole _state combobox_ ma zostawić w tabeli produkty, zgodne z wybraną wartością. W tym celu skorzystaj z komponentu [JComboBox](<https://docs.oracle.com/javase/7/docs/api/javafx/swing/JComboBox.html>).
4. **Staraj się nie mieszać modelu systemu z warstwą użytkownika.** Ogranicz interakcję pomiędzy nimi do minimum.

Uwagi i wskazówki

1. Aby oddzielić interfejs użytkownika od modelu systemu, możesz wykorzystać wzorzec projektowy Fasada.
 2. Gdy operacje są nieprawidłowe lub niezgodne z modelem systemu, zaimplementuj odpowiednie wyjątki, które wyrzucane są na warstwie modelu. ****Nie wykorzystuj wyjątków typu Runtime!**** Więcej info na ten temat [tutaj](<https://docs.oracle.com/javase/tutorial/essential/exceptions/runtime.html>).
 3. Wyjątki można obsługiwać przy pomocy komponentu JOptionPane. Przeglądaj dokumentację lub odpowiednie źródła.
 4. Aby komponenty dostosowywały się do rozmiaru okna i nie rozjeżdżały się, wykorzystaj [Layouty](<https://examples.javacodegeeks.com/desktop-java/swing/java-swing-layout-example/>).
- Godnymi uwagi na pewno będą [GridLayout](<https://examples.javacodegeeks.com/desktop-java/awt/gridlayout/java-gridlayout-example/>)
oraz [GridBagLayout](<https://examples.javacodegeeks.com/desktop-java/swing/java-swing-gridbaglayout-example/>).
5. Wykorzystaj IDE do projektowania interfejsu użytkownika. Pozwala układać komponenty przy pomocy myszki oraz udostępnia interfejs do zarządzania Layoutami! Jest to bardzo przydatne, szczególnie dla GridBagLayout'u.
 6. Stwórz klasę DataGenerator, w której uzupełnione zostaną dane, niezbędne do weryfikacji poprawności oprogramowania. Możesz wykorzystać w tym miejscu wzorzec projektowy Singleton.

Przykładowe pytania teoretyczne

1. Maven i Ant. Czym są, dlaczego i jak się ich używa?
2. Swing vs AWT
3. Layouty w Swing – po co je używać, jakie problemy rozwiązują? Jakie znasz layouty i do czego służą.
4. Obsługa zdarzeń Komponentów w Swing – ActionListener
5. SwingUtilities.invokeLater
6. Po co używamy Modeli w Swingu? Na przykładzie ListModel.
7. Idea architektury MVC. Jak wygląda MVC w Swing?
8. <https://examples.javacodegeeks.com/desktop-java/swing/java-swing-application-example/>
9. Jak możesz uniezależnić implementację UI od logiki aplikacji? Podaj przykład.