PAOiM : lab03 – UI

Zadania

Głównym zadaniem jest **stworzenie UI** dla kodu z **lab02**.
Należy pamiętać, że inne funkcje będą dla zwykłego użytkownika, a inne dla admina. Proszę o rozróżnienie tych dwóch opcji!
Admin może np. wprowadzać zmiany w danych (dodawanie, usuwanie, edycja), natomiast użytkownik może te dane przeglądać, sortować, prosić o kontakt itp.
Staraj się nie mieszać modelu systemu z warstwą użytkownika. Ogranicz interakcję pomiędzy nimi do minimum.

1. **Zaprojektuj interfejs graficzny** do zarządzania stadniną koni – 3 moduły: do logowania się, dla admina i dla klienta.
2. **Zaimplementuj obsługę interfejsu graficznego** wedle następujących reguł:
   - Interfejs składa się z dwóch list: stadniny (_list of stables) oraz koni (_list of horses in selected stable).
     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 stadnin jak i dla koni. Więcej informacji nt. AbstractTableModel na [StackOverflow] (https://stackoverflow.com/questions/9845800/abstracttablemodel-tutorial).
   - Po wybraniu stadniny (zaznaczeniu go na liście stadnin) wyświetla się dostępna w nim lista koni.
   - Zaznaczona stadnina lub koń ma zostać usunięty po naciśnięciu odpowiedniego guzika (_remove stable_ lub _remove horse_).
   - Po naciśnięciu przycisków dodaj stadninę (_add stable_) oraz dodaj konia (_add new horse_), wprowadzając odpowiednie parametry za pomocą komponentu [JOptionPane] (https://docs.oracle.com/javase/7/docs/api/javax/swing/JOptionPane.html).
     Alternatywnie, przycisk może dodać wiersz do tabeli a wprowadzenie wartości odbywa się poprzez edycję w tabeli.
   - Po naciśnięciu przycisku _sort stables by current load_ ma zostać wykonane sortowanie obiektów względem maksymalnego obciążenia. **Uwaga**: przekazuj referencje w odpowiedni sposób, wówczas taka operacja to wywołanie jednej metody.
   - Pole _filter textbox_ służy do wprowadzania nazwy konia. Po naciśnięciu klawisza enter, w tabeli mają zostać wyświetlone konie zgodne z wprowadzonym tekstem.
   - Pole _state combobox_ ma zostawić w tabeli konie, zgodne z wybraną wartością. W tym celu skorzystaj z komponentu [JComboBox] (https://docs.oracle.com/javase/7/docs/api/javax/swing/JComboBox.html).
3. Dla każdej metody stwórz **testy jednostkowe** przy użyciu biblioteki JUnit (https://github.com/junit-team/junit5/wiki). Testy zamknij w odpowiednich TestSuite. Wykorzystaj mechanizmy wbudowane w IDE do uruchomienia testów i prezentacji wyników: proszę, aby pokrycie kodu było co najmniej na poziomie 70%
4. Zaimplementuj **własne klasy wyjątków** i wykorzystaj je w najbardziej narażonych na błędy miejscach. Nie wykorzystuj wyjątków typu Runtime!

Przykładowe pytania teoretyczne

_____UI_____

1. Swing vs AWT
2. Layouty w Swing – po co je używać, jakie problemy rozwiązują? Jakie znasz layouty i do czego służą.
3. Obsługa zdarzeń Komponentów w Swing – ActionListener
4. SwingUtilities.invokeLater
5. Po co używamy Modeli w Swingu? Na przykładzie ListModel.
6. Idea architektury MVC. Jak wygląda MVC w Swing?
7. Czym jest MVC? Rola i sposób implementacji w JavaFX.
8. Różnice między JavaFX i Swing.
9. Zmiany w JavaFX od JDK11.
10. Jak możesz uniezależnić implementację UI od logiki aplikacji? Podaj przykład.

_____Wyjątki_____

11. Własne typy wyjątków, hierarchia dziedziczenia wyjątków
12. Do czego służy klasa Object i jakie ma zastosowanie ?

_____Testy_____

13. Jakie dwa rodzaje testów wyróżniamy i czym się one cechują (testy automatyczne i manualne)
14. Czym są testy jednostkowe i do czego służą?
15. Czym jest TestSuite w JUnit? [Przykład](https://howtodoinjava.com/junit5/junit5-test-suitesexamples/)
16. Czym jest Mock Objects? Gdzie się go stosuje?
17. Na czym polega Test fixture w JUnit. Jakich adnotacji możemy użyć.
18. Na czym polega Test Driven Development (TDD)?
19. Czym jest pokrycie kodu (ang. Code coverage)?
20. Jak zbudowane są testy JUnit? (adnotacje i nazwy metod)

Uwagi i wskazówki do UI

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 JOptionPanel. Przeglądnij 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/desktopjava/awt/gridlayout/java-gridlayout-example/)
   oraz [GridBagLayout](https://examples.javacodegeeks.com/desktop-java/swing/java-swinggridbaglayout-example/).
5. Wykorzystaj IDE do projektowania interfejsu użytkownika. Pozwala układać komponenty przy pomocy myszki oraz udostępnia iterfejs 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_.

Materiały do testów:

1. [Testowanie kodu w Intellij Idea] (https://www.jetbrains.com/help/idea/configuringtesting-libraries.html)
2. [Analiza pokrycia testami] (https://www.jetbrains.com/help/idea/code-coverage.html)
3. [Pisanie testów jednostkowych] (https://junit.org/junit5/docs/current/userguide/#writing-tests)
4. [Wykonywanie testów jednostkowych] https://junit.org/junit5/docs/current/userguide/#running-tests)

Materiały SWING

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-swinglayout-example/)
3. [Lista eventów w Swingu] (https://docs.oracle.com/javase/7/docs/api/javax/swing/event/packagesummary.html)
4. [Przykład obsługi evntów w Swingu] (https://www.tutorialspoint.com/swing/swing_event_handling.htm)
5. [Przykład obsługi eventów w Swingu] (https://stackoverflow.com/questions/19122514/handling-events-in-java-swing)
6. [Jak korzystać z AbstractTableModel?] (https://stackoverflow.com/questions/9845800/abstracttablemodel-tutorial)
7. [Wykorzystanie refleksji do pobrania properties] (https://stackoverflow.com/questions/8524011/java-reflection-how-can-i-get-the-all-gettermethods-of-a-java-class-and-invoke)

Materiały JavaFX

1. [JavaFX in IntelliJ] (https://www.jetbrains.com/help/idea/javafx.html)
2. JavaFX Scene Builder:
   [Scene Builder in IntelliJ] (https://www.jetbrains.com/help/idea/opening-fxml-files-injavafx-scene-builder.html
   [FXML and SceneBuilder](https://www.vojtechruzicka.com/javafx-fxml-scene-builder/)
3. [JavaFX Tutorial] (http://tutorials.jenkov.com/javafx/index.html)
4. [Tutorialspoint JavaFX] (https://www.tutorialspoint.com/javafx/index.htm)
5. [Zetcode blog] (https://www.tutorialspoint.com/javafx/index.htm)
6. Events in JavaFX:
   https://www.tutorialspoint.com/javafx/javafx_event_handling.htm
   http://www.java2s.com/Tutorials/Java/JavaFX/1100__JavaFX_Events.htm
7. [JavaFX UI controlls] (https://www.tutorialspoint.com/javafx/javafx_ui_controls.htm)
8. Controllers: https://www.educba.com/javafx-controller/
9. [Bind controller with app] (https://stackoverflow.com/questions/33881046/how-to-connect-fx-controllerwith-main-app)
10. [FXML controller example] (https://examples.javacodegeeks.com/desktop-java/javafx/fxml/javafx-fxmlcontroller-example/)