Sprawozdanie z Projektu z przedmiotu

„Event Driven Programming”

Prowadzący: por. mgr inż. Michał Sobolewski

Wykonał: Igor Irzycki

Grupa: WCY20IJ1S1

Opis aplikacji:

Aplikacja Pokedex to narzędzie dla fanów Pokemonów, które umożliwia przeglądanie informacji o różnych gatunkach Pokemonów oraz tworzenie i zarządzanie zespołami Pokemonów.

Funkcje aplikacji:

1. Przeglądanie informacji o Pokemonach: Aplikacja umożliwia użytkownikom przeglądanie szczegółowych informacji o różnych gatunkach Pokemonów, takich jak statystyki, typy, ewolucje itp. Dane te są pobierane z zewnętrznego API (np. PokeAPI).
2. Wyszukiwanie Pokemonów: Użytkownicy mogą wyszukiwać konkretnych Pokemonów na podstawie ich nazw lub numerów. Aplikacja prezentuje wyniki wyszukiwania w formie listy lub kafelków z podglądem podstawowych informacji o każdym Pokemonie.
3. Tworzenie i zarządzanie zespołami Pokemonów: Użytkownicy mogą tworzyć swoje własne zespoły Pokemonów, dodając je do wybranych zespołów. Aplikacja umożliwia dodawanie i usuwanie Pokemonów z zespołów oraz wyświetlanie ich szczegółowych statystyk.
4. Wizualizacja statystyk: Aplikacja oferuje wykresy i grafiki, które prezentują statystyki Pokemonów w zespołach. Użytkownicy mogą łatwo porównywać moc, obronę, prędkość itp. różnych Pokemonów.
5. Konfiguracja połączenia z bazą danych: Aplikacja umożliwia użytkownikom skonfigurowanie połączenia z bazą danych, w której można przechowywać informacje o zespołach Pokemonów. Dane logowania do bazy danych są przechowywane w pliku konfiguracyjnym.

Wykorzystane technologie:

1. Java: Aplikacja Pokedex jest napisana w języku Java, który jest popularnym językiem programowania wykorzystywanym do tworzenia aplikacji desktopowych i webowych. Java zapewnia szerokie wsparcie dla bibliotek i narzędzi, co ułatwia rozwój i utrzymanie aplikacji.
2. JavaFX: Do tworzenia interfejsu użytkownika w aplikacji Pokedex wykorzystano bibliotekę JavaFX. JavaFX jest częścią standardowej biblioteki Java, która umożliwia tworzenie graficznych interfejsów użytkownika (GUI). Dostarcza bogate komponenty, takie jak przyciski, etykiety, tabele itp., oraz umożliwia tworzenie atrakcyjnych i interaktywnych interfejsów.
3. REST API (np. PokeAPI): Aplikacja Pokedex korzysta z zewnętrznego API (Application Programming Interface) do pobierania danych o Pokemonach. W przykładzie może być wykorzystane PokeAPI, które udostępnia informacje o różnych gatunkach Pokemonów. API pozwala na pobieranie danych, takich jak statystyki, typy, ewolucje i wiele innych, które są wykorzystywane do prezentacji szczegółowych informacji o Pokemonach w aplikacji.
4. Apache Commons: W aplikacji Pokedex wykorzystano bibliotekę Apache Commons, która dostarcza wiele narzędzi i funkcji pomocnych podczas programowania w języku Java. Na przykład, można wykorzystać klasę FileUtils do operacji na plikach, klasę StringUtils do manipulacji tekstami, a klasę IOUtils do operacji wejścia/wyjścia.
5. Guava (Google Core Libraries for Java): Biblioteka Guava, stworzona przez Google, oferuje zestaw narzędzi i funkcji ułatwiających programowanie w języku Java.
6. Maven: Maven jest narzędziem do zarządzania zależnościami, budowania i konfiguracji projektów Java. Umożliwia łatwe zarządzanie bibliotekami, kompilację kodu, tworzenie pakietów aplikacji i wiele innych operacji.
7. Baza danych MySQL: Aplikacja Pokedex może wykorzystywać bazę danych do przechowywania informacji o zespołach Pokemonów.

Wykorzystane biblioteki:

Wszystkie wykorzystane biblioteki zostały odpowiednio skonfigurowane i dodane do pliku **pom.xml** jako zależności (dependencies) w projekcie Maven. Dzięki temu, biblioteki zostały automatycznie pobrane i zaimportowane do projektu, co umożliwia korzystanie z ich funkcji i narzędzi.

Obraz zawierający tekst, zrzut ekranu, menu, Czcionka

Opis wygenerowany automatycznie

Obraz zawierający tekst, zrzut ekranu, Czcionka, menu

Opis wygenerowany automatycznie

Obraz zawierający tekst, zrzut ekranu, menu, Czcionka

Opis wygenerowany automatycznieObraz zawierający tekst, zrzut ekranu, menu, Czcionka

Opis wygenerowany automatycznie

Działanie aplikacji:

Strona główna aplikacji wygląda następująco:

Obraz zawierający tekst, zrzut ekranu, Czcionka, logo

Opis wygenerowany automatycznie

Po wybraniu opcji „Search Pokemon” zostaje wyświetlona następująca scena:

Obraz zawierający tekst, zrzut ekranu, diagram, design

Opis wygenerowany automatycznie

Po kliknięciu opcji „Load All” wyświetli się lista wszystkich Pokemonów. Za pomocą komponentu ComboBox można filtrować pokemony według danej Generacji. Opcjonalnie można wyszukać Pokemona po jego numerze w Pokedexie oraz po jego nazwie. Po wybraniu Pokemona z listy pojawia się jego zdjęcie, wyświetlają się statystyki, informacje o nim, a także jego umiejętności oraz linia ewolucyjna.

Załadowanie wszystkich pokemonów:

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Ikona komputerowa

Opis wygenerowany automatycznie

Filtrowanie generacji:

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Ikona komputerowa

Opis wygenerowany automatycznie

Wybranie pokemona z listy:

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Ikona komputerowa

Opis wygenerowany automatycznie

Wyszukanie pokemona po jego nazwie:

Obraz zawierający tekst, oprogramowanie, Ikona komputerowa, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

Wyszukanie pokemona po jego numerze w pokedexie:

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Ikona komputerowa

Opis wygenerowany automatycznie

Możemy za pomocą przycisku „Go Back” cofnąć się do menu głównego i wybrać kolejną scenę „Teams Management”:

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Strona internetowa

Opis wygenerowany automatycznie

Możemy tu stworzyć kilka drużyn jednocześnie i uzupełniać je w dowolnej kolejności. Kiedy wybierzemy z listy stworzoną drużynę i klikniemy przycisk „Add to Database” to dana drużyna zostanie zapisana w bazie danych. Możemy usuwać pokemony z drużyny w fazie tworzenia drużyny.

Po kliknięciu przycisku „Show Teams” zostanie otwarte dodatkowe okno, w którym możemy przeglądać zapisane w bazie danych drużyny:

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, komputer

Opis wygenerowany automatycznie

Możemy tu wyświetlić daną drużynę używając ComboBox, a także wyświetlić danego Pokemona z drużyny poprzez wybranie go z listy. Po wybraniu pokemona zostają wyświetlone jego statystyki na wykresie kolumnowym. Możemy tu również usunąć wybrane pokemony w drużynie, a nawet usunąć całą drużynę z bazy danych.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Strona internetowa

Opis wygenerowany automatycznie

Kliknięcie przycisku „Delete Pokemon”:

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Strona internetowa

Opis wygenerowany automatycznie

Score Card:

1. Technologia Java SE.

2. Aplikacja GUI

Aplikacja Pokedex została w pełni wyposażona w graficzny interfejs użytkownika (GUI) i została napisana z wykorzystaniem frameworka JavaFX oraz narzędzia Scene Builder.

3. Wielowątkowość.

4.Asynchroniczność.

W powyższym kodzie funkcji **handleListAllButtonClick**, używamy asynchroniczności do pobrania listy wszystkich Pokemonów z API. Wykorzystujemy klasę **CompletableFuture** z pakietu **java.util.concurrent**, która pozwala nam wykonywać operacje asynchroniczne i przetwarzać wyniki w momencie, gdy są gotowe.

1. Tworzymy **CompletableFuture** przy użyciu metody **supplyAsync**, która przyjmuje lambdę jako argument. Ta lambda reprezentuje kod do wykonania w tle. W tym przypadku, wykonujemy żądanie HTTP do API Pokemon, aby pobrać listę wszystkich Pokemonów. Ten kod zostanie uruchomiony w oddzielnym wątku, nie blokując wątku głównego interfejsu użytkownika.
2. W bloku **thenAcceptAsync**, który jest wywołany po zakończeniu operacji asynchronicznej, przetwarzamy otrzymane dane w formacie JSON. Ponownie, używamy lambdy do przekazania kodu do wykonania.
3. Wewnątrz lambdy, sprawdzamy, czy otrzymane dane JSON są niepuste (**json != null**). Jeśli tak, używamy **Platform.runLater**, aby zapewnić, że operacje modyfikujące interfejs użytkownika są wykonywane w głównym wątku aplikacji. W przeciwnym razie, w przypadku błędu lub braku danych, nie wykonujemy żadnych operacji.
4. Wewnątrz bloku **Platform.runLater**, czyścimy listę **listViewAllPokemon** i dodajemy do niej nazwy Pokemonów pobrane z danych JSON. Ta operacja jest bezpieczna do wykonania w głównym wątku, ponieważ korzystamy z metody **runLater**, która umożliwia bezpieczne aktualizacje interfejsu użytkownika.

Dzięki temu podejściu, mamy możliwość pobierania danych z API w tle, nie blokując wątku głównego interfejsu użytkownika. Aby uniknąć blokady interfejsu i zapewnić płynność działania aplikacji, operacje pobierania danych są wykonywane asynchronicznie, a aktualizacje interfejsu są przekazywane do głównego wątku.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie

Opis wygenerowany automatycznie

5. Zdarzenia (własne zdarzenia).

Wymaganie dotyczące "Zdarzeń" w projekcie odnosi się do wykorzystania gotowych mechanizmów zdarzeń dostarczanych przez używane biblioteki lub frameworki. Przykładem takiego mechanizmu zdarzeń może być JavaFX Event System, które jest wbudowane w JavaFX i umożliwia obsługę różnych zdarzeń w interfejsie użytkownika. Przykład spełnienia tego wymagania w projekcie może obejmować obsługę zdarzenia kliknięcia przycisku.



Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

6. Wykorzystanie danych pobranych z dwóch zewnętrznych usług webowych (dozwolone było użycie tylko jednej usługi).

Aplikacja korzysta z jednego API pobranego z zewnętrznej usługi webowej, a mianowicie z Poke API. Poke API dostarcza szczegółowe informacje o Pokemonach, takie jak ich nazwy, typy, statystyki, umiejętności itp.

7. Bazy danych (JDBC, JPA)

Aplikacja wykorzystuje bazę danych MySQL do przechowywania i zarządzania danymi związanych z utworzonymi zespołami Pokemonów.

8. Config Properties.

W moim projekcie wykorzystałem plik konfiguracyjny **config.properties**, aby przechowywać dane logowania do bazy danych MySQL. Plik **config.properties** jest plikiem tekstowym, który zawiera klucze i wartości w formacie **klucz=wartość**, gdzie klucz to nazwa konfiguracji, a wartość to odpowiadająca mu wartość.

9. Własne 2 komponenty graficzne.

Aplikacja korzysta z wielu komponentów graficznych, takich jak guziki, wykres, comboBox, ListView, ImageView, TextArea itp.

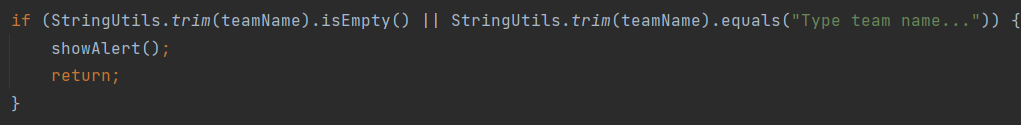
10. Dodatkowo punktowane wykorzystanie reflection API.

W moim projekcie wykorzystałem Reflection API w kontekście klasy **FXMLLoader**, która jest częścią JavaFX i służy do wczytywania plików FXML.

W aplikacji, podczas tworzenia interfejsu użytkownika z wykorzystaniem plików FXML, użyłem **FXMLLoader** do dynamicznego ładowania plików FXML i tworzenia odpowiednich obiektów kontrolerów.

11. Wykorzystanie bibliotek Apache Commons oraz magistrali zdarzeniowych.

Biblioteka Apache Commons została wykorzystana do edycji zmiennych typu String, np.:



12. Zastosowanie wzorców programowania obiektowego i zdarzeniowego.

13. Zastosowanie wzorców projektowych z serii strukturalnych i behawioralnych.

14. Wymaga się całkowitego autorstwa kodów źródłowych.