

Uniwersytet Przyrodniczo – Humanistyczny
w Siedlcach

Kierunek Informatyka

Imię i nazwisko studenta
Rafał Geresz gr. 1

Temat zadania
Samoobsługowa stacja benzynowa

Praca wykonana pod kierunkiem
mgr inż. Anna Kołkowicz

Siedlce, 2023

1. Specyfikacja wymagań

Program pozwala na wybór 1 z trzech stanowisk, wybór paliwa dostępnego na stanowisku (po wybraniu pokazuje się cena), następnie wpisanie ilości paliwa, którą chcemy zatankować. Po tym wszystkim, program pokazuje kwotę do zapłaty, a po naciśnięciu przycisku zatankuj rozpoczyna się tankowanie.

Przy zakończeniu pracy z programem dane są zapisywane do pliku, a przy ponownym uruchomieniu dane są wczytywane z pliku.

2. Opis klas

GasStation

Główna klasa przechowująca dane w programie. Składa się z trzech obiektów klasy Station (Stanowisko). Ta klasa może mieć zainicjowany tylko jeden obiekt dzięki wzorcowi projektowemu Singleton. Odpowiada również za zapisywanie oraz odczytywanie z pliku danu o stanowiskach.

Station

Klasa Station jest abstrakcyjną klasą publiczną, która implementuje interfejs Serializable. Posiada trzy prywatne właściwości (name, error, working) reprezentowane przez klasy StringProperty i BooleanProperty z biblioteki JavaFX.

Konstruktor klasy Station przyjmuje parametr name i inicjalizuje właściwości name, error i working za pomocą klas SimpleStringProperty i SimpleBooleanProperty. Wartością domyślną dla właściwości working jest true.

Klasa Station udostępnia metody dostępne do właściwości name, error i working, które zwracają odpowiednie obiekty StringProperty lub BooleanProperty. Dodatkowo udostępnia metody getName(), getError() i getWorking(), które zwracają wartości tych właściwości. Metody setName(), setError() i setWorking() służą do ustawiania nowych wartości dla odpowiednich właściwości.

Klasa Station jest klasą abstrakcyjną, jest rozszerzana przy pomocy klas StationDieselAndGas i StationLPG. Klasa ta deklaruje również abstrakcyjną metodę refuel(Double amount, String type), która musi być zaimplementowana przez klasy dziedziczące. Metoda ta może zgłaszać trzy wyjątki: PaperException, DistributorFailureException i SpendingChangeException.

StationDieselAndGas

Klasa StationGasAndDiesel jest publiczną klasą, która dziedziczy po klasie Station i implementuje interfejs Serializable. Klasa ta reprezentuje stację paliwową obsługującą zarówno paliwo diesel, jak i benzynę.

Posiada cztery prywatne właściwości: dieselAmount, dieselPrice, gasAmount i gasPrice, które są reprezentowane przez klasy DoubleProperty z biblioteki JavaFX.

Konstruktor klasy StationGasAndDiesel przyjmuje parametry name, dieselAmount, dieselPrice, gasAmount i gasPrice. Wywołuje również konstruktor klasy nadrzędnej Station za pomocą super(name), aby zainicjalizować właściwość name.

W klasie StationGasAndDiesel zaimplementowana jest metoda refuel(Double amount, String type) jako przesłonięcie metody abstrakcyjnej z klasy Station. Metoda ta obsługuje proces tankowania paliwa. Jeśli podany typ paliwa to "Diesel", zmniejsza ilość paliwa diesel o podaną ilość. Jeśli podany typ to "Benzyna", zmniejsza ilość paliwa benzynowego o podaną ilość. Dodatkowo, metoda może zgłaszać wyjątki PaperException, DistributorFailureException i SpendingChangeException. Wewnątrz metody sprawdzane jest również, czy występuje awaria dystrybutora, co jest symulowane przez losowanie liczby i sprawdzanie jej podzielności przez 5.

Klasa StationGasAndDiesel udostępnia metody dostępne do właściwości dieselAmount, dieselPrice, gasAmount i gasPrice, które zwracają odpowiednie obiekty DoubleProperty. Dodatkowo, udostępnia metody getDieselAmount(), addDiesel(), getDieselPrice(), setDieselPrice(), getGasAmount(), addGas(), getGasPrice(), setGasPrice(), które pozwalają na pobranie i ustawienie wartości tych właściwości. Metody addDiesel() i addGas() również mogą zgłaszać wyjątek BelowZeroException, jeśli podana ilość jest ujemna.

Klasa StationGasAndDiesel posiada również metodę toString(), która zwraca łańcuch znaków reprezentujący stan obiektu. Zawiera nazwę stacji, ilość paliwa diesel, cenę paliwa diesel, ilość paliwa benzynowego oraz cenę paliwa benzynowego.

StationLPG

Klasa StationLPG jest publiczną klasą, która dziedziczy po klasie Station i implementuje interfejs Serializable. Reprezentuje stację paliwową obsługującą gaz LPG.

Klasa StationLPG posiada dwie prywatne właściwości: gasPrice i gasAmount, które są reprezentowane przez klasy DoubleProperty z biblioteki JavaFX.

Konstruktor klasy StationLPG przyjmuje parametry name, gasAmount i gasPrice. Wywołuje również konstruktor klasy nadrzędnej Station za pomocą super(name), aby zainicjalizować właściwość name.

W klasie StationLPG zaimplementowana jest metoda refuel(Double amount, String type) jako przesłonięcie metody abstrakcyjnej z klasy Station. Metoda ta obsługuje proces tankowania gazu LPG. Jeśli podany typ paliwa to "Gaz", zmniejsza ilość gazu o podaną ilość.

Klasa StationLPG udostępnia metody dostępne do właściwości gasPrice i gasAmount, które zwracają odpowiednie obiekty DoubleProperty. Dodatkowo, udostępnia metody getGasPrice(), setGasPrice(), getGasAmount(), addGas() i setGasAmount(), które pozwalają na pobranie i ustawienie wartości tych właściwości. Metody addGas() i setGasAmount() mogą zgłaszać wyjątek BelowZeroException, jeśli podana ilość jest ujemna.

Klasa StationLPG posiada również metodę toString(), która zwraca łańcuch znaków reprezentujący stan obiektu. Zawiera nazwę stacji, ilość gazu LPG oraz cenę gazu LPG.

ErrorLabel

Klasa `ErrorLabel` jest publiczną klasą, która rozszerza klasę `HBox`. Reprezentuje etykietę błędu składającą się z tekstu oraz przycisku naprawy.

Klasa `ErrorLabel` posiada dwie prywatne właściwości: `lbl` typu `Label` i `btn` typu `Button`.

Konstruktor klasy `ErrorLabel` przyjmuje parametr `text`, który jest używany do utworzenia etykiety (`Label`) o podanym tekście. Etykieta jest stylizowana za pomocą atrybutów CSS, takich jak kolor tekstu, waga czcionki, rozmiar czcionki i minimalna szerokość.

Następnie konstruktor tworzy przycisk (`Button`) o napisie "Napraw". Przycisk również jest stylizowany za pomocą atrybutów CSS, takich jak kolor tła, kolor tekstu, waga czcionki i rozmiar czcionki.

Oba elementy (`lbl` i `btn`) są dodawane do kontenera typu `HBox`, który jest kontenerem poziomym.

Klasa `ErrorLabel` udostępnia metody `getLbl()` i `getBtn()`, które zwracają odpowiednio etykietę i przycisk. Pozwala to na dostęp do tych elementów z zewnątrz klasy, na przykład w celu dodania ich do sceny lub manipulacji nimi.

ErrorDialog

Klasa `ErrorDialog` jest publiczną klasą, która rozszerza klasę `Alert`. Reprezentuje okno dialogowe z komunikatem o błędzie.

Konstruktor klasy `ErrorDialog` przyjmuje parametr `text`, który jest używany jako treść komunikatu błędu.

W konstruktorze wywoływany jest konstruktor klasy nadrzędnej `Alert` z argumentem `AlertType.ERROR`, co ustawia typ okna dialogowego na "błąd". Następnie ustawiane są tytuł okna, nagłówek (tekst podsumowujący błąd) oraz treść komunikatu błędu za pomocą metod `setTitle()`, `setHeaderText()` i `setContentText()`.

Klasa `ErrorDialog` nie dodaje żadnych dodatkowych funkcji ani właściwości. Służy do wyświetlania prostego okna dialogowego z informacją o błędzie.

PaperException

Klasa `PaperException` jest publiczną klasą, która dziedziczy po klasie `Exception`. Reprezentuje wyjątek związany z papierem.

Konstruktor klasy `PaperException` przyjmuje parametr `message`, który jest używany jako wiadomość wyjątku. Wywołuje również konstruktor klasy nadrzędnej `Exception` za pomocą `super(message)`, aby zainicjalizować wiadomość wyjątku.

Klasa `PaperException` nie dodaje żadnych dodatkowych funkcji ani właściwości. Służy do tworzenia i przechwytywania wyjątków związanych z papierem w aplikacji.

Pozostałe klasy wyjątków wyglądają analogicznie, więc nie będę ich opisywał.

ClientPanelTab1Controller

Klasa ClientPanelTab1Controller jest kontrolerem dla interfejsu użytkownika, który obsługuje pierwszą zakładkę panelu klienta.

Pola klasy oznaczone adnotacją @FXML reprezentują elementy interfejsu użytkownika powiązane z kontrolerem, takie jak pola tekstowe (TextField), przyciski (Button), przyciski opcji (RadioButton), etykiety (Label) itp.

Pole station reprezentuje obiekt klasy StationGasAndDiesel, który jest stacją paliwową dla zakłádki.

Metoda initialize jest wywoływana podczas inicjalizacji kontrolera i jest używana do ustawienia początkowego stanu interfejsu użytkownika. W metodzie tej pole station jest inicjalizowane przy użyciu metody GasStation.getInstance().getStation(0), a następnie wiązane jest tekstowe właściwość etykiety priceLbl z właściwością gasPriceProperty obiektu station.

Metoda amountTyped jest wywoływana, gdy użytkownik wprowadza wartość do pola tekstowego amountTxt. Metoda ta parsuje wprowadzoną wartość na Double i oblicza do zapłaty na podstawie wybranej opcji paliwa (dieselRBtn lub gasRBtn). Jeśli wprowadzona wartość jest niepoprawna lub mniejsza od zera, odpowiedni komunikat jest wyświetlany w etykiecie toPayLbl.

Metoda fuelTypeChange jest wywoływana, gdy zmienia się wybór opcji paliwa (dieselRBtn lub gasRBtn). Metoda ta aktualizuje tekstową właściwość etykiety priceLbl zgodnie z wybraną opcją paliwa.

Metoda refuel jest wywoływana po kliknięciu przycisku "Refuel". Sprawdza, czy stacja jest czynna (station.getWorking()), jeśli nie, wyświetla okno dialogowe z komunikatem o błędzie. Następnie parsuje wprowadzoną wartość z pola tekstowego amountTxt i sprawdza, czy jest wystarczająca ilość paliwa do zatankowania. Jeśli nie, zostaje zgłoszony wyjątek DistributorFailureException. Wywoływana jest metoda pay do przeprowadzenia płatności, a następnie metoda refuel z obiektu station w celu zatankowania odpowiedniej ilości paliwa.

Metoda pay jest używana do przeprowadzenia płatności. Oblicza kwotę do zapłaty na podstawie wprowadzonej ilości paliwa i ceny paliwa. Wyświetla okno dialogowe z pytaniem o podanie kwoty zapłaty. Po wprowadzeniu kwoty sprawdza, czy jest wystarczająca do pokrycia całej kwoty do zapłaty. Jeśli nie, aktualizuje stan stacji i ponownie wywołuje pay. Jeśli kwota jest wystarczająca, wywołuje showReceipt w celu wyświetlenia paragonu.

Metoda showReceipt jest używana do wyświetlenia paragonu. Generuje losowy numer, a następnie sprawdza, czy liczba jest podzielna przez 5. Jeśli tak, ustawia stan stacji na nieczynny i zgłasza wyjątek PaperException lub SpendingChangeException. Następnie wyświetla okno dialogowe z informacjami o paragonie.

Metoda getFuelType zwraca wybrany rodzaj paliwa (gasRBtn lub dieselRBtn) jako ciąg znaków.

Metoda getFuelPrice zwraca cenę wybranego rodzaju paliwa (gasRBtn lub dieselRBtn) jako ciąg znaków.

Pozostałe kontrolery działają podobnie, więc nie będę ich opisywał.

3. Diagram klas



4. Kod programu

Kod programu, dokumentacja oraz diagramy klas znajdują się na GitHubie:
<https://github.com/Rgeresz/SB>.

5. Przykładowe dane i wyniki

Przykład 1

Dane: podanie ilości litrów, naciśnięcie Tankuj oraz podanie kwoty wystarczającej do zapłaty.

Wynik: Pokazanie paragonu, wydanie reszty oraz zataknowanie LUB błąd mówiący o braku papieru, braku drobnych lub awarii dystrybutora.

Przykład 2

Dane: podanie ilości litrów, naciśnięcie Tankuj oraz podanie kwoty niewystarczającej do zapłaty.

Wynik: pokazanie błędu o zbyt małej ilości pieniędzy oraz pokazanie okna z prośbą o podanie kwoty.

Przykład 3

Dane: podanie nowej ceny w panelu pracownika i naciśnięcie przycisku Zmień cenę.

Wynik: zmiana ceny

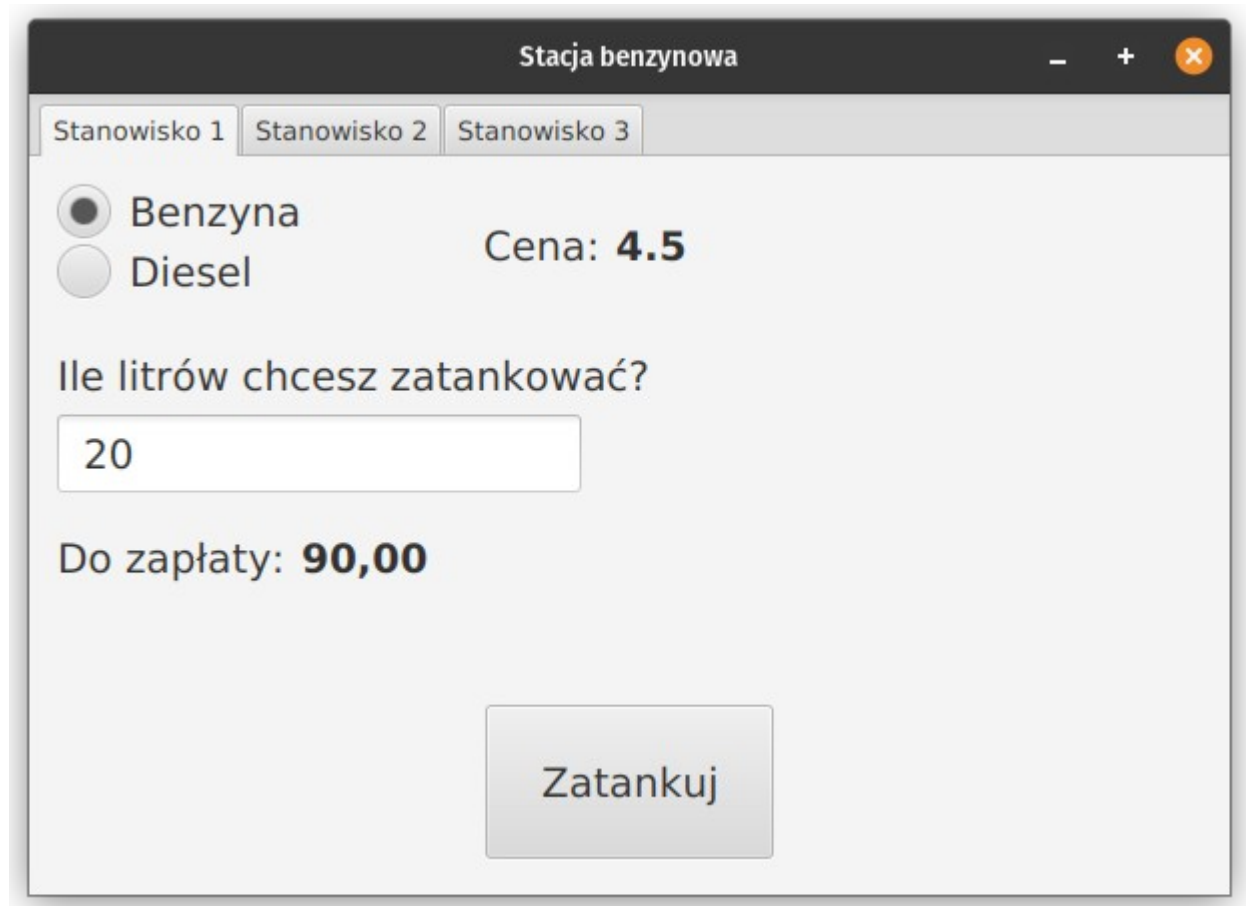
Przykład 3

Dane: podanie niepoprawnie nowej ceny w panelu pracownika i naciśnięcie przycisku Zmień cenę.

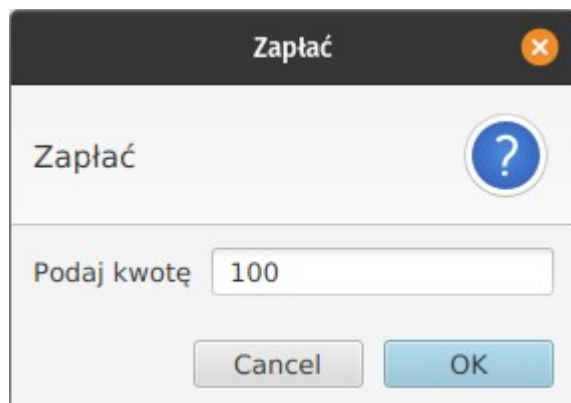
Wynik: pokazanie komunikatu o błędzie z odpowiednią informacją

6. Instrukcja dla użytkownika

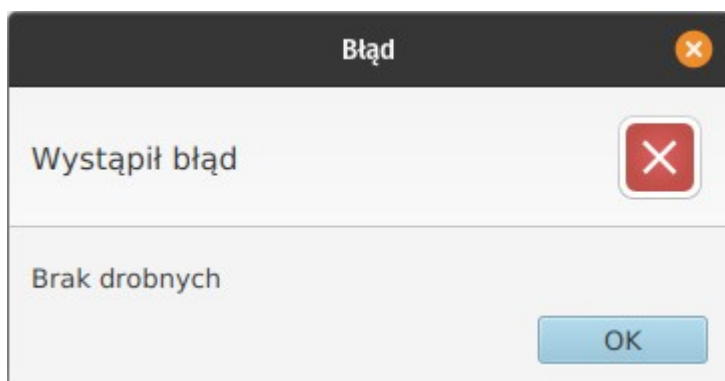
Po uruchomieniu programu pokazuje się okno z trzema zakładkami symbolizującymi trzy stanowiska. W dwóch pierwszych mamy do wyboru benzynę oraz diesla w trzecim tylko gaz. Obok nazwy paliwa pokazuje się jego cena. Pod mamy pole do wpisania ilości paliwa, którą chcemy kupić. Pod tym polem mamy informację, ile zapłacimy za tą ilość paliwa. Gdy jesteśmy gotowi do zakupu, naciskamy Zatankuj.



Po naciśnięciu Zatankuj pojawia się okno z prośbą o podanie kwoty. Wpisujemy kwotę wystarczającą do zapłaty za wpisaną ilość paliwa i wciskamy OK.



Po naciśnięciu OK pokazuje się albo okno o błędzie np. takie

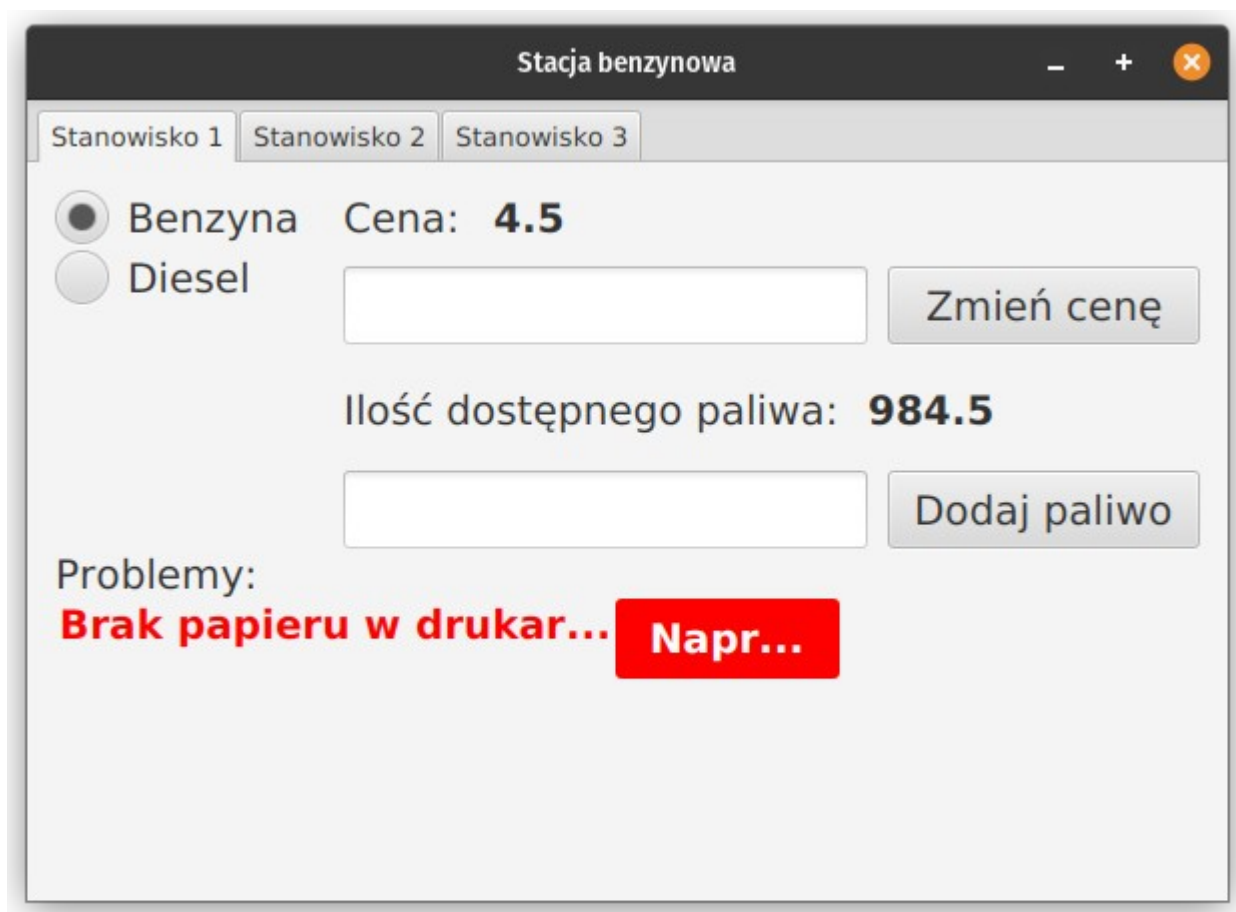


lub okno z paragonem



Po naciśnięciu przycisku Zatankuj możemy również zobaczyć błąd „Stanowisko nieczynne”. Wtedy należy wezwać pracownika stacji lub skorzystać z innego stanowiska.

Aby dostać się do panelu pracownika należy w panelu klienta nacisnąć Ctrl+a oraz podać hasło „admin”. W tym panelu analogicznie do panelu klienta mamy trzy zakładki odpowiadające stanowiskom. Możemy tam zmienić cenę wpisując w pole obok przycisku zmień cenę nową cenę po czym zatwierdzając tym przyciskiem. Tak samo postępujemy dodając paliwo. Na dole pojawiają się problemy na danym stanowisku i możemy je usunąć używając czerwonego przycisku Napraw.



Stacja benzynowa

Stanowisko 1 Stanowisko 2 Stanowisko 3

☒ Benzyna Cena: **4.5**

☐ Diesel

Zmień cenę

Ilość dostępnego paliwa: **984.5**

Dodaj paliwo

Problemy:

Brak papieru w drukar...

Napr...

Aby powrócić do panelu klienta ponownie naciskamy Ctrl+a.