AKADEMIA GÓRNICZO-HUTNICZA

Dokumentacja do projektu

Design and implement classes for a flight reservation system

z przedmiotu

Języki programowania obiektowego

Elektronika i Telekomunikacja 3 rok

Kacper Klepacz

poniedziałek 14:40

prowadzący: Rafał Frączek

16 stycznia 2021

1. Opis projektu

Projekt rezerwacji biletów lotniczych wraz z graficznym interfejsem, jest dedykowany dla małej rozwijającej się linii lotniczej, oferującej loty z trzech lotnisk w Krakowie do trzech krajów za granicą, po wyborze lotniska z którego lecimy oraz miejsca w które się kierujemy, musimy dokonać wyboru dnia oraz godziny oferowanego połączenia. Bilety są oferowane jako „last-minute" i wylot jest zawsze w przyszłym tygodniu od rezerwacji. Na koniec otrzymujemy wypełniony bilet uwzględniając nasze dane i poprzednie wybory.

2. Project description

Flight reservation project with graphic user interface dedicated for small growing airline. Program allows user to choose one from three departure airports, and three destinations to countries abroad from each airport. User is choosing only the day and time of the flight because they are "last-minute" flights so you can buy ticket for next week only. In the end user receives the ticket filled with its data entered previously.

3. Instrukcja użytkownika

Na start otrzymujemy menu wyboru czynności, jaką chcemy wykonać, poruszamy się strzałkami lub myszką, dostępne są rezerwacja nowego biletu oraz podejrzenie zrobionej rezerwacji(w przypadku braku poprzedniej rezerwacji otrzymamy informację o braku biletu do wyświetlenia), po przejściu do rezerwacji biletu dostaniemy menu wyboru lotniska, z którego chcemy odlecieć, następnie po wyborze najbardziej nam odpowiadającego wybieramy kierunek lotu. Po wybraniu tych dwóch kluczowych informacji, pozostaje nam wybór godziny, dnia oraz przejście do wprowadzenia potrzebnych danych do rezerwacji(należy pamiętać, że wpisując dane poruszamy się po polach strzałkami klawiatury). Zależy nam na jak najszybszej rezerwacji, dlatego ograniczamy potrzebne dane do imienia, nazwiska oraz adresu email. Naciskamy „print a ticket" i naszym oczom ukazuję się bilet zawierający informacje o locie, nasze dane oraz przydzielone miejsce i klasę w samolocie.

4. Kompilacja

Projekt zapewnia kompilację w systemie Windows, przy użyciu Visual Studio, dodatkowo wymagana jest kofiguracja oraz dodanie biblioteki SFML. Należy również pamiętać o plikach źródłowych projektu oraz pliku .sln i plikach .dll.

5. Pliki źródłowe

Projekt składa się z następujących plików źródłowych:

- *Tickets_booking.cpp* – plik zawierający główną funkcję `main()` programu operującą na wszystkich zawartych klasach,
- *fsm*.h – plik zawierający deklarację automatu sterującego przełączaniem się miedzy oknami w programie,
- *Ticket.h*, *Ticket.cpp* – deklaracja oraz implementacja klasy `Ticket` zawierającej funkcje do poruszania się po oknie biletu,
- *TicketDisplay.h*, *TicketDisplay.cpp* – deklaracja oraz implementacja klasy `TicketDisplay` zawierająca wyświetlenie okna zawierającego bilet wraz z uzupełnionymi danymi,
- *Menu.h*, *Menu.cpp* – deklaracja oraz implementacja klasy `Menu` zawierającej funkcje do poruszania się po pierwszym oknie,
- *MainMenuDisplay.h*, *MainMenuDisplay.cpp* – deklaracja oraz implementacja klasy `MainMenuDisplay` zawierająca wyświetlenie okna do poruszania się oraz wybór oferowanych opcji,

- *DataWindow.h*, *DataWindow.cpp* – deklaracja oraz implementacja klasy `DataWindow` zawierającej funkcje do poruszania się po oknie z wprowadzaniem danych,
- *EnterDataDisplay.h*, *EnterDataDisplay.cpp* – deklaracja oraz implementacja klasy `EnterDataDisplay` zawierająca wyświetlenie okna do poruszania się, wybór oferowanych opcji, wprowadzenie danych oraz ich zapisanie do wykorzystania w późniejszych etapach programu,
- *ChooseDestination.h*, *ChooseDestination.cpp* – deklaracja oraz implementacja klasy `ChooseDestination` zawierającej funkcje do poruszania się po oknie wyboru kierunku lotu,
- *DestinationDisplay.h*, *DestinationDisplay.cpp* – deklaracja oraz implementacja klasy `DestinationDisplay` zawierająca wyświetlenie okna do poruszania się oraz wybór oferowanych opcji kierunku lotu,
- *DateTime.h*, *DateTime.cpp* – deklaracja oraz implementacja klasy `DateTime` zawierającej funkcje do poruszania się po oknie wyboru godziny i dnia lotu,
- *DateOfFlightDisplay.h*, *DateOfFlightDisplay.cpp* – deklaracja oraz implementacja klasy `DateOfFlightDisplay` zawierająca wyświetlenie okna do poruszania się oraz wybór oferowanych opcji dnia i godziny,
- *ChooseAirport.h*, *ChooseAirport.cpp* – deklaracja oraz implementacja klasy `ChooseAirport` zawierającej funkcje do poruszania się po oknie wyboru lotniska, z którego chcemy lecieć,
- *AirportDisplay.h*, *AirportDisplay.cpp* – deklaracja oraz implementacja klasy `AirportDisplay` zawierająca wyświetlenie okna do poruszania się oraz wybór oferowanych opcji lotnisk,

6. Zależności

W projekcie wykorzystano następujące dodatkowe biblioteki:

- SFML – biblioteka na której bazuje całe GUI projektu: https://www.sfml-dev.org/.

7. Opis klas

W tym punkcie należy umieścić opis wszystkich stworzonych w projekcie klas. Należy podać do czego służy dana klasa oraz informację o jej publicznych metodach. Opcjonalnie można załączyć fragmenty kodu źródłowego. Na przykład:

W projekcie utworzono następujące klasy:

- `TicketDisplay` – zarządzenie danymi zmienionymi w pozostałych klasach oraz pokazanie ich na ekranie.
  - `void draw(sf::RenderWindow& window, FSM& fsm, int airportsymbol, int& isticketavailable, int randseat, sf::Sprite& background, sf::Sprite& logo)-rysuje zmienne, bilet oraz grafikę na ekranie, po której się poruszamy, którą czytamy i dokonujemy wyboru kolejnych okien,`
  - `Ticket ticket` – tworzymy obiekt z klasy `Ticket` którego użyjemy do wywołania potrzebnych funkcji,
  - `sf::Sprite button2` - przycisk wyświetlany na ekranie,
  - `sf::Sprite tickett` – bilet wyświetlany na ekranie,
  - `sf::Sprite kremowka` – symbol fikcyjnego lotniska wyświetlany na ekranie,

- `sf::Sprite hutasymbol` – symbol fikcyjnego lotniska wyświetlany na ekranie ,
- `sf::Sprite balicesymbol` - symbol lotniska wyświetlany na ekranie,
- `sf::Text *name` – dana zawierająca imię podane w poprzednich oknach,
- `sf::Text *surname` – dana zawierająca nazwisko podane w poprzednich oknach,
- `sf::Text *email` – dana zawierająca email podany w poprzednich oknach,
- `sf::Text *destinationprint` – dana zawierająca kierunek wybrany w poprzednich oknach,
- `sf::Text seatprint` – dana zawierająca wygenerowane miejsce w samolocie do wyświetlenia,
- `sf::Text seattext` – dana zawierająca prefix miejsce w samolocie,
- `sf::Text classtext` – dana zawierająca klasę miejsce w samolocie,
- `sf::Text *flightno` – dana zawierająca numer lotu wybrany w poprzednim oknie,
- `sf::Text *timetprint` – dana zawierająca godzinę lotu podaną w poprzednim oknie,
- `sf::Text *dayprint` – dana zawierająca dzień lotu podany w poprzednim oknie,
- `int randseat` – dana zawierająca generację miejsca w samolocie.

- `Ticket` – poruszanie się po menu z biletem.
  - `void draw(sf::RenderWindow& window)`–rysuje tekst na ekranie, po którym się poruszamy i dokonujemy wyboru kolejnych okien,
  - `void MoveUp`()-poruszanie się po menu w górę,
  - `void MoveDown()`- poruszanie się po menu w dół,
  - `int GetPressedItem()` – sprawdzenie którą opcję wybierzemy.
- `Menu` – poruszanie się po początkowym menu.
  - `void draw(sf::RenderWindow& window)`–rysuje tekst na ekranie, po którym się poruszamy i dokonujemy wyboru kolejnych okien,
  - `void MoveUp()`–poruszanie się po menu w górę,
  - `void MoveDown()`- poruszanie się po menu w dół,
  - `int GetPressedItem()` – sprawdzenie którą opcję wybierzemy.
- `MainMenuDisplay` – Wyświetlanie GUI menu początkowego oraz operowanie na nim.
  - `void draw()`–rysuje zmienne, przyciski oraz grafikę na ekranie, po której się poruszamy, którą czytamy i dokonujemy wyboru kolejnych okien,
  - `Menu menu` – tworzymy obiekt z klasy `Ticket` którego użyjemy do wywołania potrzebnych funkcji,
  - `sf::Sprite button1,2,3` – przycisk wyświetlany na ekranie,
  - `sf::Text infoprint` – informacja o tym, że nie ma biletu do wyświetlenia.
- `fsm` – Deklaracja automatu ze stanami okien.

- `EnterDataDisplay` – Klasa odpowiedzialna za wprowadzanie danych takich jak imię i nazwisko i przekazanie ich do innych klas.
  - `void draw(sf::RenderWindow& window, FSM& fsm, int& randseat, sf::Sprite& background, sf::Sprite& logo)` –rysuje zmienne, przyciski oraz grafikę na ekranie, po której się poruszamy, którą czytamy i dokonujemy wyboru kolejnych okien,
  - `DataWindow dataWindow` – obiekt klasy tworzony na potrzeby użycia funkcji,
  - `sf::Sprite button1,2,3,4,5,6,7,8` – przyciski wyświetlane na ekranie,
  - `sf::Text *name` – dana zawierająca imię,
  - `sf::Text *surname` – dana zawierająca nazwisko,
  - `sf::Text *email` – dana zawierająca email,
  - `sf::String nameInput` – dana pobierająca z klawiatury imię,
  - `sf::String surnameInput` – dana pobierająca z klawiatury nazwisko,
  - `sf::String emailInput` – dana pobierająca z klawiatury email.
- `DestinationDisplay` – Klasa odpowiedzialna za wyświetlenie opcji kierunku lotu i operacje na ich wyborze.
  - `void draw(sf::RenderWindow& window, FSM& fsm, int& checkdest, sf::Sprite& background, sf::Sprite& logo)` –rysuje zmienne, przyciski oraz grafikę na ekranie, po której się poruszamy, którą czytamy i dokonujemy wyboru kolejnych okien,
  - `ChooseDestination chooseDestination` – obiekt klasy tworzony na potrzeby użycia funkcji,
  - `sf::Sprite button1,2,3,4` – przyciski wyświetlane na ekranie,
  - `sf::Sprite italyflag, germany flag, maldivesflag` – flagi printowane na ekranie,
  - `sf::Text destinationprint` – tekst pobierający wybrany kierunek lotu,
  - `sf::Text montime` – tekst pobierający wybraną godzinę lotu w poniedziałek,
  - `sf::Text wedtime` – tekst pobierający wybraną godzinę lotu w środę,
  - `sf::Text fritime` – tekst pobierający wybraną godzinę lotu w piątek.
- `DateTime` – poruszanie się po oknie z danymi.
  - `void draw(sf::RenderWindow& window)` –rysuje tekst na ekranie po którym się poruszamy,
  - `void MoveUp()` – poruszanie się po menu w górę,
  - `void MoveDown()` – poruszanie się po menu w dół,
  - `int GetPressedItem()` – sprawdzenie którą opcję wybierzemy.

- `DateOfFlightDisplay` – Klasa odpowiedzialna za wybór i wyświetlenie wybor dnia i godziny lotu.
  - `void draw(sf::RenderWindow& window, FSM& fsm, int checkdest, sf::Sprite& background, sf::Sprite& logo)` – rysuje zmienne, przyciski oraz grafikę na ekranie, po której się poruszamy, którą czytamy i dokonujemy wyboru kolejnych okien,
  - `DateTime dateTime` – obiekt klasy tworzony na potrzeby użycia funkcji,
  - `sf::Sprite button1,2,3,4` – przyciski wyświetlane na ekranie,
  - `sf::Text *montime` – tekst zawierający wybraną godzinę lotu w poniedziałek,
  - `sf::Text *wedtime` – tekst zawierający wybraną godzinę lotu w środę,
  - `sf::Text *fritime` – tekst zawierający wybraną godzinę lotu w piątek.
  - `sf::Text dayprint` – dana zawierająca dzień lotu,
  - `sf::Text timetprint` – dana zawierająca godzinę lotu.
- `DataWindow` – Klasa odpowiedzialna za poruszanie się po oknie wpisywania danych.
  - `void draw(sf::RenderWindow& window)`–rysuje tekst na ekranie po którym się poruszamy,
  - `void MoveUp()` – poruszanie się po menu w górę,
  - `void MoveDown()` – poruszanie się po menu w dół,
  - `int GetPressedItem()` – sprawdzenie którą opcję wybierzemy.
- `ChooseDestination` – Klasa odpowiedzialna za poruszanie się po oknie wyboru kierunku.
  - `void draw(sf::RenderWindow& window)`–rysuje tekst na ekranie po którym się poruszamy,
  - `void MoveUp()` – poruszanie się po menu w górę,
  - `void MoveDown()` – poruszanie się po menu w dół,
  - `int GetPressedItem()` – sprawdzenie którą opcję wybierzemy.
- `ChooseAirport` – Klasa odpowiedzialna za poruszanie się po oknie wyboru lotniska.
  - `void draw(sf::RenderWindow& window)`–rysuje tekst na ekranie po którym się poruszamy,
  - `void MoveUp()` – poruszanie się po menu w górę,
  - `void MoveDown()` – poruszanie się po menu w dół,
  - `int GetPressedItem()` – sprawdzenie którą opcję wybierzemy.
- `AirportDisplay` – Klasa odpowiedzialna za wybór i wyświetlenie wyboru lotniska z którego lecimy.
  - `void draw(sf::RenderWindow& window, FSM& fsm, int& airportsymbol, sf::Sprite& background, sf::Sprite& logo)` – rysuje zmienne, przyciski oraz grafikę na ekranie, po której się poruszamy, którą czytamy i dokonujemy wyboru kolejnych okien,
  - `ChooseAirport chooseAirport` – obiekt klasy tworzony na potrzeby użycia funkcji,

- `sf::Sprite button1,2,3,4` – przyciski wyświetlane na ekranie,
- `sf::Text flightno` – dana zawierająca numer identyfikujący lot.

8. Zasoby

Jeśli projekt wykorzystuje jakieś dodatkowe zasoby jak na przykład pliki z danymi tekstowymi, pliki obrazów itp. to w tym punkcie należy je wyszczególnić. W przypadku plików tekstowych konieczne jest opisanie struktury takiego pliku. Jeśli w projekcie nie ma żadnych zasobów to piszemy słowo „brak”. Na przykład:

W projekcie wykorzystywane są następujące pliki zasobów:

- `kremowka.png` – plik zawierający zdjęcie symbolu fikcyjnego lotniska w Wadowicach,
- `hutnictwosymbol.png` – plik zawierający zdjęcie symbolu fikcyjnego lotniska w dzielnicy Nowa Huta,
- `balicelotnisko.jpg` – plik zawierający zdjęcie symbol lotniska w Balicach,
- `airlineslogo.PNG` – plik zawierający zdjęcie loga linii lotniczej,
- `backgroundplane.jpg` – plik zawierający tło GUI,
- `italyflag.jpg, maldivesflag.jpg, germanyflag.png` – pliki zawierające zdjęcia flag krajów, do których lecimy,
- `button.png` – plik z teksturą przycisków po których poruszamy się w GUI,
- `ticket.jpg` – plik ze zdjęciem szablonu biletu linii lotniczej,
- `Minecraft-Regular.otf` – plik z czcionką użytą w projekcie.

9. Dalszy rozwój i ulepszenia

- Wysyłanie potwierdzenia na maila w celu zapłacenia za rezerwację
- Wybór większej ilości lotnisk
- Możliwość wyboru miejsca w samolocie
- Rezerwacja wielu biletów na raz

10. Inne

Brak.