Projektowanie obiektowe 2024L

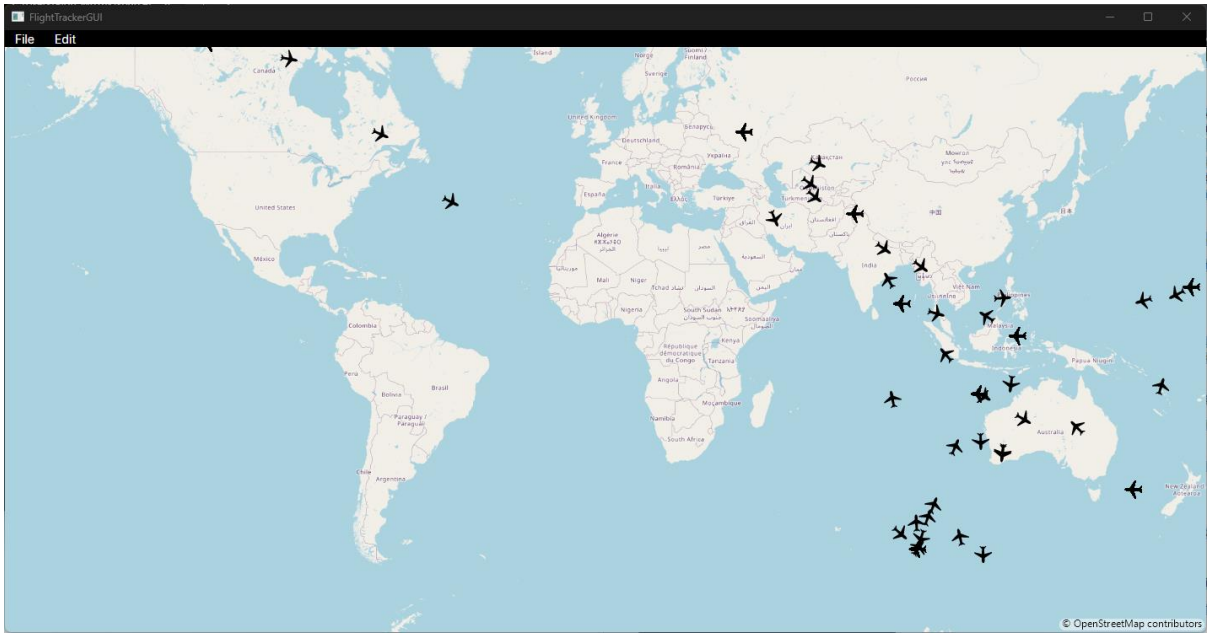
Etap 3 – Integracja z GUI

Wstęp

W tym etapie należy dodać synchronizację danych uzyskanych z serwera bądź pliku tekstowego z aplikacją okienkową (Graphical User Interface – GUI).

Aplikacja z interfejsem została przygotowana i jest możliwa do zaimportowania do projektu za pomocą pakietu nuget. Do wyświetlenia aplikacji nie jest konieczna znajomość programowania w środowisku graficznym, gdyż wszystkie potrzebne metody do interakcji z kontrolkami okienka zostały wyciągnięte do osobnych funkcji.

Wygląd aplikacji



Na okno aplikacji składają się mapa świata z naniesionymi na nią ikonkami reprezentującymi pozycję samolotów w danej chwili. Na mapie wyświetlane są wszystkie samoloty, dla których istnieje lot w przechowywanej bazie danych. Pozycje samolotów są stale odświeżane. Ruch jest na tyle niewielki, że zaleca się przybliżenie mapy w celu zaobserwowania ruchu samolotu.

Wyświetlana mapa może zostać przybliżona za pomocą scrolla na myszy.

Komunikacja z aplikacją

Pakiet nuget udostępnia uproszczone wrappery na funkcje wymagane do wyświetlenia okienka i zaktualizowania pozycji samolotów.

W tym celu należy skorzystać z funkcji:

- namespace FlightTrackerGUI:
  - → `void Runner.Run()` powinna zostać wywołana w celu uruchomienia aplikacji.
  - → `void Runner.UpdateGUI(FlightsGUIData flightsGUIData)` powinna zostać wywołana w celu wysłania nowej listy lotów do okna aplikacji w celu odświeżenia wyświetlanych danych.

Celem zadania jest napisanie konwersji między danymi lotów z klasy Flight a oczekiwanymi danymi FlightGUI. Należy również obliczyć pozycję samolotów w świecie (długość i szerokość geograficzną) oraz rotację w przestrzeni 2D wyświetlanej mapy.

Pozycja samolotu powinna zostać obliczona na podstawie lokalizacji (długości i szerokości geograficznej) lotniska początkowego i końcowego – tzn. długości i szerokości geograficznej ich pozycji. Wartość ta powinna być wartością zinterpolowaną w zależności od obecnego czasu względem czasu wylotu i przylotu danego samolotu. Pozycja samolotów powinna być aktualizowana co sekundę. Kierunek lotu samolotu obliczany jest na podstawie obecnej i poprzedniej lokalizacji.

Do obliczenia poprawnego kąta, może przydać się transformacja długości i szerokości geograficznej na współrzędne X, Y na mapie, i można się w tym celu posłużyć funkcją z dostępnego w zaimportowanym nugecie pakietu MapsUI:

```
SphericalMercator.FromLonLat
```

Struktura klasy FlightGUIData oraz FlightGUI

Dane przyjmowane przez funkcję UpdateGUI są w postaci obiektu klasy:

```
public class FlightsGUIData
{
private List<FlightGUI> flightsData;
virtual public int            GetFlightsCount()
virtual public UInt64         GetID(int index);
virtual public WorldPosition  GetPosition(int index);
virtual public double         GetRotation(int index);
}
```

Obiekty wewnętrznej listy `flightsData` mają natomiast postać obiektów klasy:

```
public class FlightGUI
{
    public UInt64            ID { get; init; }
    public WorldPosition     WorldPosition { get; init; }
    public double            MapCoordRotation { get; init; }
}
```

ID jest unikatowym identyfikatorem lotu. WorldPostion jest strukturą przechowującą szerokość i długość geograficzną. MapCoordRotation jest natomiast kątem między wektorem kierunku lotu we współrzędnych mapy a wektorem (0,1), podanym w radianach. Upraszczając, gdy podany kąt = 0, samolot jest skierowany w górę na mapie, dla kąta=PI/2 - w prawo itd.

Termin wykonania

2 tygodnie

Wszystkie pliki źródłowe muszą zostać wgrane na repozytorium git do 26.03.2024 23:59.

Projekt należy zaprezentować prowadzącemu w czasie zajęć 27.03.2024.