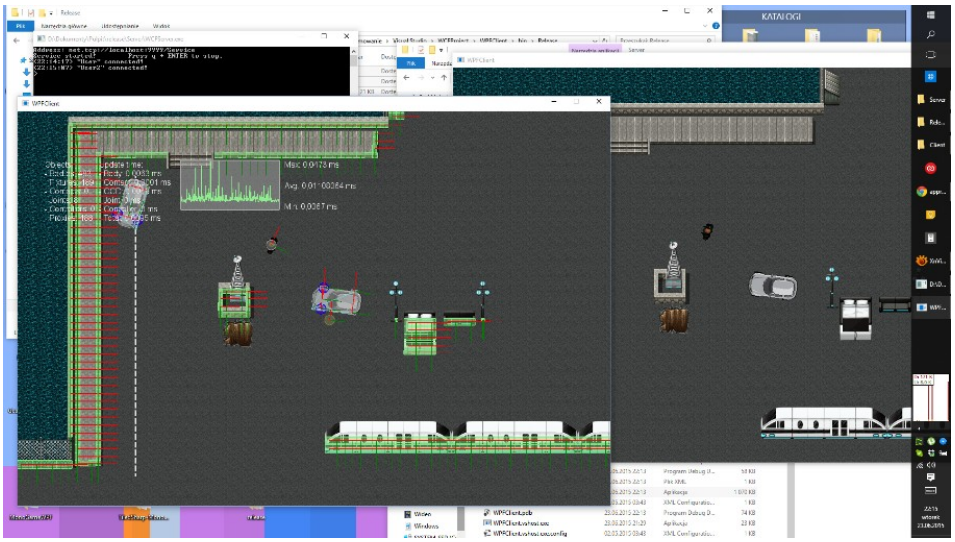
Dokumentacja projektu systemu rozproszonego „WCFProject”

Cel

Celem projektu było stworzenie gry sieciowej dla wielu graczy z wykorzystaniem dostępnych technologii tworzenia systemów rozproszonych.



Podstawowe funkcje

Projekt przewiduje zaimplementowanie podstawowych funkcji systemu rozproszonego z wyszczególnieniem:

- Łączności dwukierunkowej klient-server;
- Logów serwera dostępnych z poziomu konsoli;
- Mapy na której toczyć ma się rozgrywka;
- Oddzielnej aplikacji klienckiej;
- Serwer z możliwością uruchomienia na dowolnym systemie wspierającym technologię .NET Framework 4.5 (w tym Mono) lub jako usługa IIS;

Wykorzystane technologie i biblioteki

Programowanie implementacji umożliwił język C#, a dane przesyłane między aplikacjami i zapisywane w plikach zapisu wykorzystują format JSON. W solucji zostały zawarte technologie i biblioteki firm trzecich z wyróżnieniem technologii firmy Microsoft. Lista wszystkich użytych frameworków przedstawia się następująco:

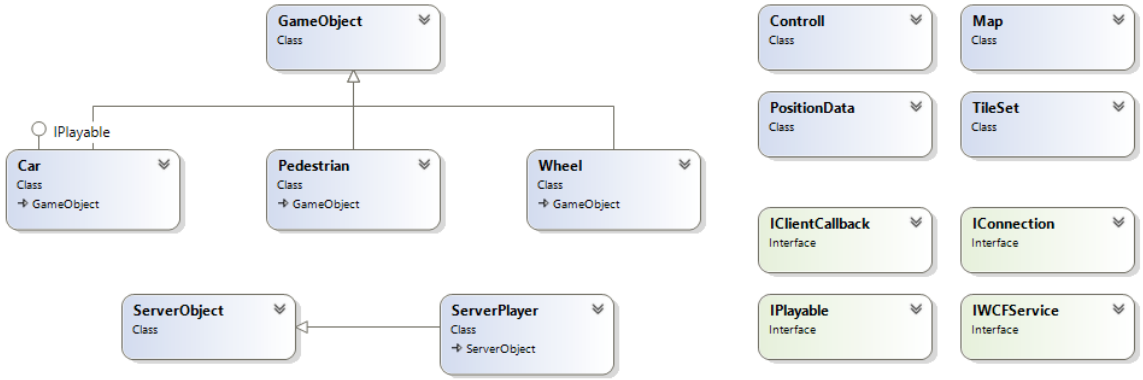
- Microsoft .NET Framework 4.5;
- Windows Presentation Foundation (WPF);
- Windows Communication Foundation (WCF);
- MonoGame;
- Farseer Physics Engine (Box2D);
- Json.NET;
- TiledSharp;
- MahApps.Metro;

Implementacja

Projekt składa się z 4 głównych części oraz bibliotek. Wybrane klasy zostały dokładnie opisane w kodzie źródłowym (w języku angielskim). W skład podstawowych elementów wchodzą:

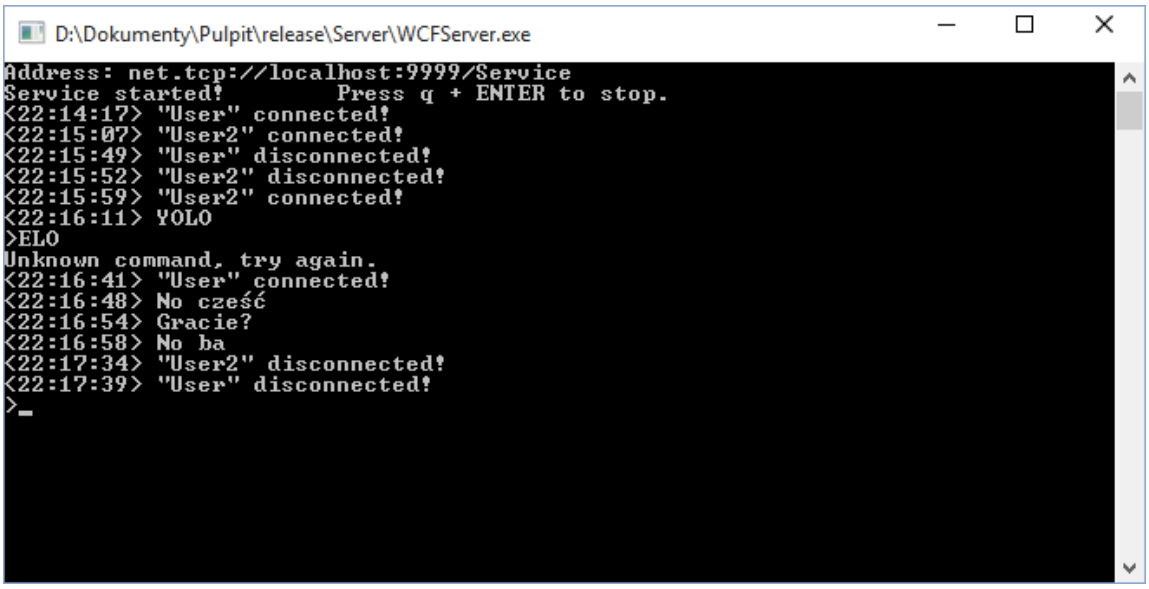
- WCFReference – część wspólna aplikacji klienckiej i serwerowej, zawiera wszystkie podstawowe obiekty i typy serializowane w tym DataContracts;
- WCFServer – aplikacja serwera;
- WPFClient – aplikacja klienta;
- GameWindow – gra będąca referencją dla aplikacji klienckiej, zawiera wszystkie elementy niezbędne do uruchomienia okna gry, na chwilę obecną nie może działać jako oddzielna aplikacja;

WCFReference



- GameObject – podstawowa klasa abstrakcji, implementuje właściwości takie jak pozycja, rotacja, numer identyfikacyjny obiektu, ciało, czy rozmiar i metody odpowiedzialne za aktualizację i rysowanie obiektu;
- Car, Pedestrian, Wheel – dziedziczące po GameObject finalne klasy wprowadzające dodatkowe właściwości i nadpisujące podstawowe metody w celu dopasowania do konkretnego typu elementu;
- ServerObject, ServerPlayer – klasy podlegające serializacji przez format JSON, przesyłają informacje z serwera do klienta i zawierają dane o dwóch podstawowych typach obiektów – bardziej ogólny Object i obiekty będące danymi zalogowanych graczy Player;
- Controll – klasa przechowująca dane wejściowe od użytkownika (klawiatura, mysz);
- PositionData – obiekt używany do przesyłania danych klient -> serwer, zawiera dane konkretnego zalogowanego gracza o jego pozycji na mapie;
- TileSet – zapewnia obsługę grafik typu kafelki, dzieli podany obiekt i dostarcza dane o położeniu kafelków;
- Map – przechowuje i deserializuje plik mapy TMX, a także dostarcza metody ułatwiające wykorzystanie biblioteki TiledSharp;
- IWCFService – interfejs niezbędny do użycia technologii WCF, informuje klienta o dostępnych do wykorzystania metodach serwera;
- IClientCallback – podobnie jak w przypadku IWCFService, jednak tym razem przechowuje informacje dotyczące dostępnych dla serwera metod klienta w komunikacji dupleks;
- IPlayable, IConnection – interfejsy pomocnicze ułatwiające implementację;

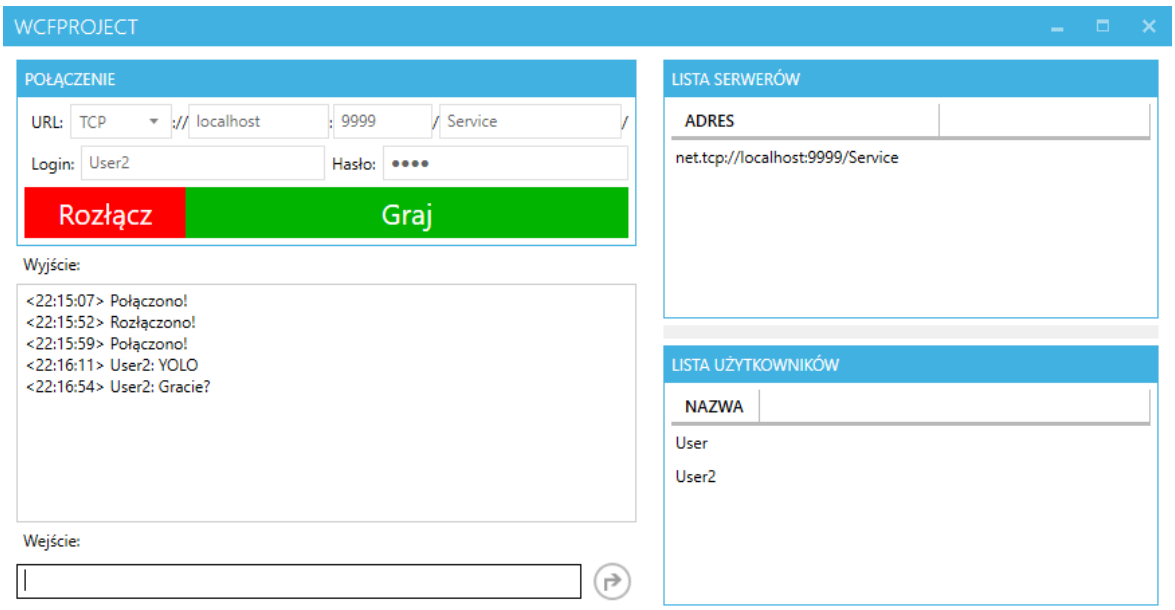
WCFServer



Prosty singleton definiujący klasę zawierającą metody dostępne dla klienta. Ponadto posiada częściowo zaimplementowany silnik fizyki, który wykrywa kolizje obiektów, tym samym pomagając zachować spójność świata gry u każdego klienta. Wszystkie obiekty obliczane po stronie klienta są również obliczane po stronie serwera zapobiegając różnicom w obliczeniach kolizji u każdego z klientów. Może zostać wykorzystany do zabezpieczenia przeciw nieuczciwym praktykom, tj. używania „hacków” modyfikujących aplikację kliencką tak, aby dało się wykonać czynności niedozwolone przez zasady gry.

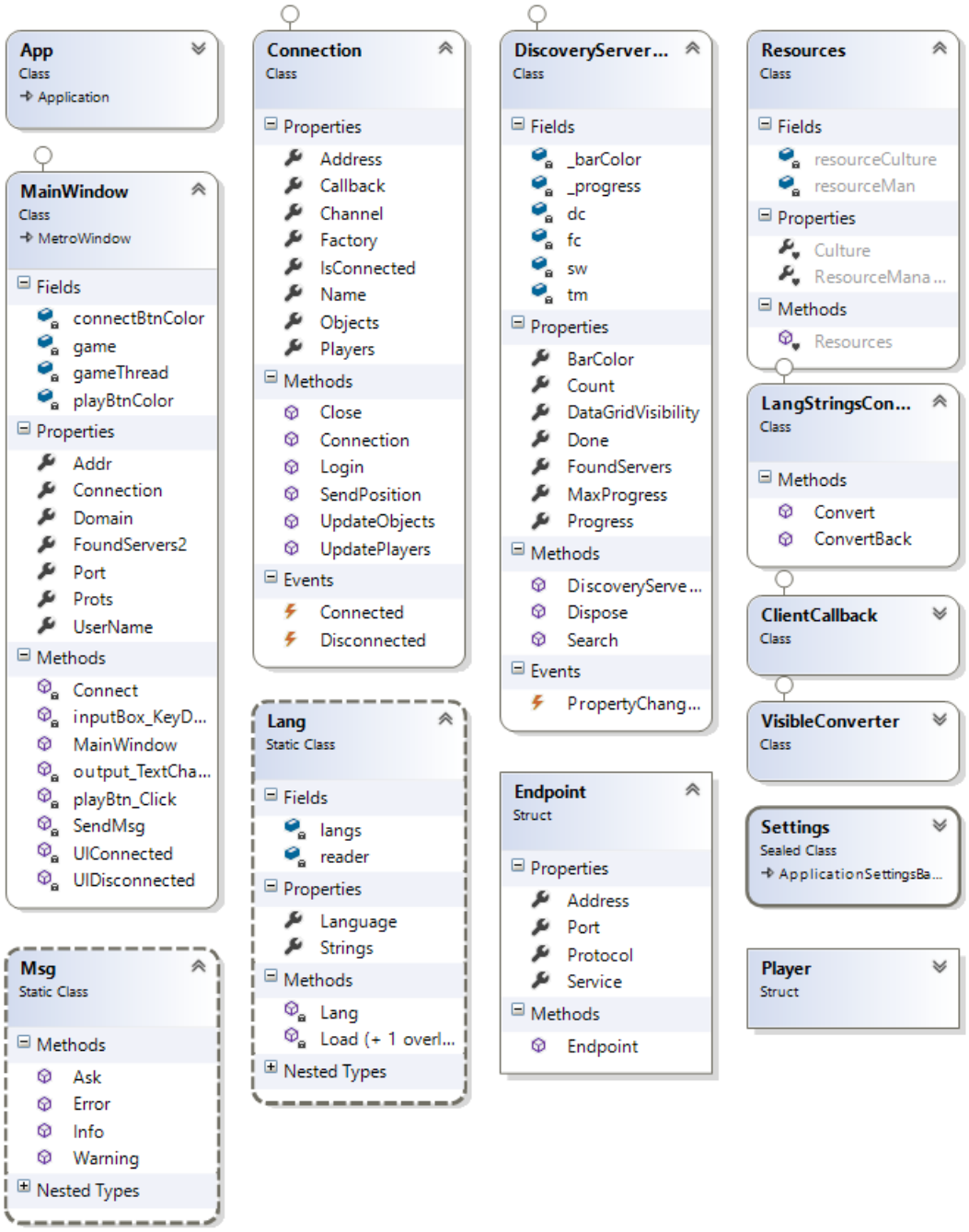
WPFClient

Wygląd



Aplikacja kliencka oparta o framework Windows Presentation Foundation zapewnia optymalny komfort tworzenie przyjaznych dla oka aplikacji okienkowych przy niezbyt dużym wysiłku. Wykorzystane zewnętrzne referencje wyglądu dodają oknu wrażenia nowoczesności i zgodności z najnowszymi standardami tworzenia aplikacji okienkowych.

Działanie

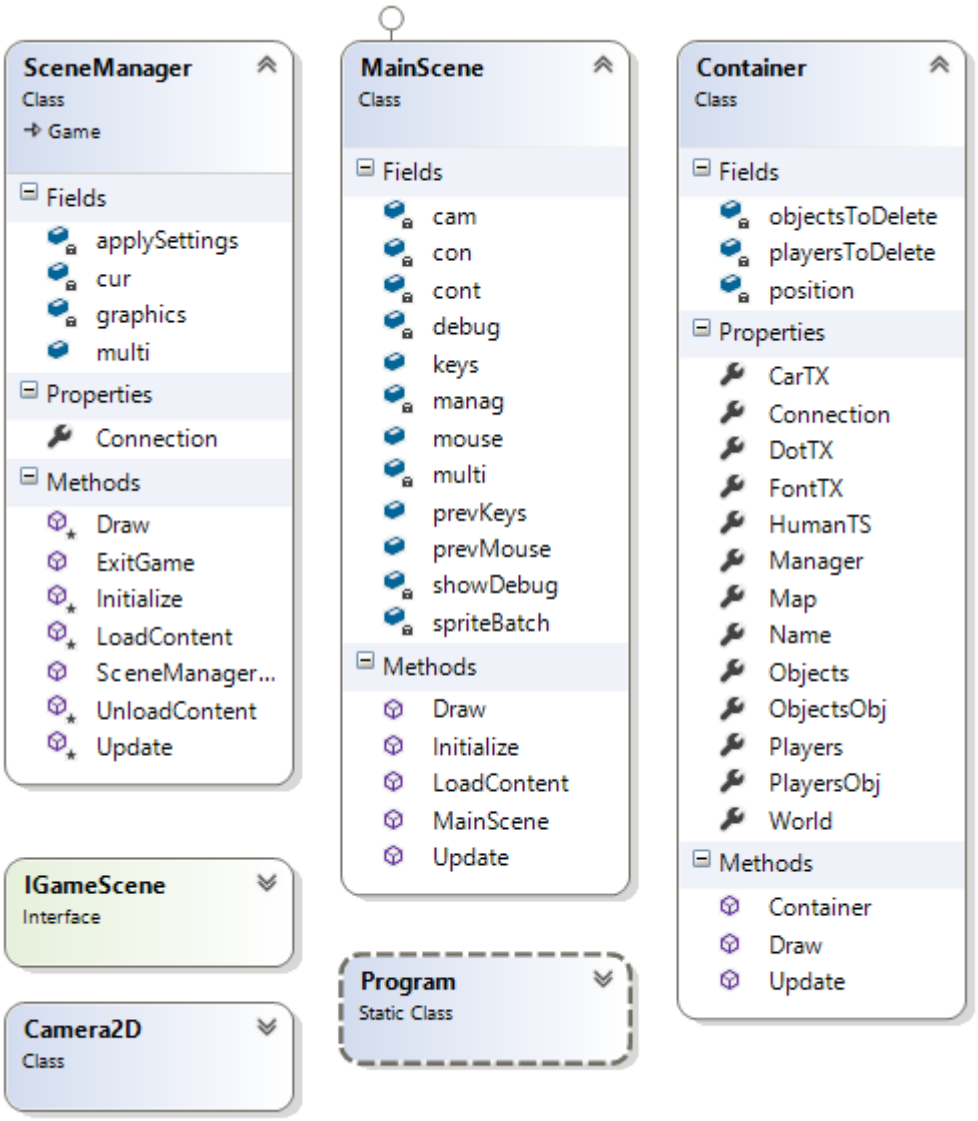


Aplikacja kliencka napisana z użyciem wzorca projektowego .NET MVVM (Model-View-ViewModel).

Zadaniem aplikacji klienckiej jest nawiązanie połączenia z aplikacją serwera i utworzenie kanału komunikacyjnego. Dodatkowym zadaniem jest informowanie użytkownika o stanie działania całego programu – statusu połączenia, dostępności serwerów, listę użytkowników. Przy użyciu okna klienta możliwe jest także wysłanie wiadomości do serwera bez potrzeby otwierania okna gry.

Po poprawnym nawiązaniu połączenia launcher pozwala na uruchomienie okna gry z już skonfigurowanym połączeniem, w trakcie rozgrywki monitoruje także stan połączenia i pokazuje na bieżąco informacje w postaci listy logów.

GameWindow



Niepozorne serce gry. Posiada wszystkie niezbędne referencje i pliki zawartości takiej jak tekstury, fonty i mapy. Klasa SceneManager zarządza działaniem wszystkich elementów i dba o ich aktualizację i rysowanie w odpowiednim czasie. Inicjalizuje także połączenie z serwerem ustanowione wcześniej przez WPFClient.

Obecnie jedyną sceną zaimplementowaną w grze jest MainScene wyświetlająca mapę gry, obiekty i graczy. To w tej klasie obliczane są również kolizje i pobierane dane wejściowe od gracza.

Container przechowuje wszystkie obiekty i referencje do obiektów gry, a także implementuje większość uniwersalnych dla scen metod, jak obliczanie świata, rysowanie obiektów, ładowanie źródeł grafik i fontów.

Camera2D to niepozorny, ale bardzo ważny obiekt implementujący, jak sama nazwa wskazuje, funkcję ruchomej kamery z możliwością przybliżania i oddalania obrazu i gładkim podążaniem za aktywnym obiektem na mapie.

Uruchomienie i obsługa

Aby uruchomić instancję serwera niezbędnego do funkcjonowania całego systemu rozproszonego można wykorzystać prekompilowane pliki z katalogu „BinaryRelease/Server”. Instancję serwera należy uruchomić z pliku „WCFServer.exe”.

Uwaga! Jeśli pojawią się błędy przy uruchamianiu instancji serwera może to być spowodowane niezainstalowaną odpowiednią wersją .NET Framework (wymagany co najmniej 4.5) lub plikami zapisu.

W przypadku problemów z uruchomieniem zalecamy usunięcie plików „ObjectsData.json” i „PlayersData.json” z katalogu „BinaryRelease/Server” i ponowne uruchomienie instancji serwera.

Instancje klienta można uruchomić z katalogu „BinaryRelease/Client” z pliku „WPFClient.exe”. Aby poprawnie przetestować działanie systemu rozproszonego należy uruchomić przynajmniej dwie instancje klienta.

Uwaga! Serwer nie pozwoli na zalogowanie się dwóm użytkownikom z tą samą nazwą co zostanie zasygnalizowane odpowiednim komunikatem, dlatego przed połączeniem drugiego klienta należy zmienić domyślną nazwę „User” na inną np. „User2”.

Po poprawnym połączeniu można uruchomić okna gry we wszystkich instancjach klienta.

Sterowanie

Gra jest sterowana przy użyciu myszy i/lub klawiatury.

Sterowanie jako pieszy (Pedestrian):

- W, S, A, D
- mysz (kierunek) + LPM (do przodu) + PPM na samochodzie (wejście/wyjście z samochodu);

Sterowanie samochodem (Car):

- W, S, A, D, Spacja (hamulec ręczny);

Inne:

- Tab (wł./wył. wyświetlanie DebugView);
- Num+ (przybliż mapę);
- Num- (oddal mapę);

Zastrzeżenia

- WCFProjekt jest systemem rozproszonym przygotowanym na zaliczenie z przedmiotu „Systemy rozproszone”;
- Aplikacja(e) posiadają/mogą posiadać wiele błędów związanych z brakiem niektórych funkcji i/lub niedokładnym dopracowaniem istniejących;
- Mapa użyta w WCFProjekt została stworzona przy użyciu programu Tiled i jest jedynie przykładem możliwej do stworzenia mapy;
- Cały projekt jest przykładem implementacji systemu rozproszonego i nie należy go traktować jako gotowy produkt;
- WCFProjekt nie jest przygotowany do codziennego użytkowania i rozpowszechniania;