Michał Miłosz, 282594

PROZE – etap 4.

Protokół sieciowy dla gry Space Invaders.

Wstęp:

Mój protokół sieciowy opisuje w jaki sposób klient łączy się z serwerem oraz komunikuje się z nim w celu pobrania/wysłania odpowiednich informacji potrzebnych w trakcie rozgrywki. Serwer przechowuje i udostępnia graczom listę najlepszych wyników oraz plik konfiguracyjny.

Ogólny opis pracy w tym protokole:

- Klient pobiera dane konfiguracyjne aplikacji,
- Klient pobiera z serwera listę najlepszych wyników,
- Klient pobiera dane o konkretnym poziomie,
- Klient zapisuje wysyła do serwera informacje o wyniku obecnego gracza.

Protokół ma charakter tekstowy. Jest oparty o mechanizm socketów i protokół TCP. Wszystkie wysyłane dane mają formę tekstową. Wszelkie odpowiedzi serwera, niezależnie od ich objętości, są wysyłane w postaci jednej linii odpowiedzi, a parametry w poszczególnych zdaniach są odseparowane od siebie spacją.

Przykładowe połączenie:

W przykładowym scenariuszu zakładamy, że połączenie następuje pomiędzy jednym klientem i jednym serwerem. Klient ma nazwę C.

1. Klient wysyła do serwera polecenie login, które oznacza chęć nawiązania połączenia, po czym klient czeka na odpowiedź serwera

   `C:login =>S`

2. Serwer otrzymuje od klienta na ustalonym porcie żądanie LOGIN z nazwą klienta i wysyła do niego wiadomość LOGGEDIN <nr_w_wektorze>, gdzie nr_w_wektorze to wartość całkowitoliczbowa dodatnia.

   `S:loggedin <nr_w_wektorze> => C`

3. Klient pobiera parametry konfiguracyjne potrzebne do uruchomienia aplikacji – wysyła żądanie GET_CONFIG i oczekuje na odpowiedź serwera.

   `C:GET_CONFIG =>S`

4. Serwer odbiera żądanie GET_CONFIG po czym wysyła parametry konfiguracyjne w odpowiedniej kolejności:

   `S: GET_CONFIG <WspółrzędnaX> <WspółrzędnaY> <ileLeveli>(…) =>C`

5. Klient postanawia, że chce pobrać z serwera listę najlepszych wyników. Wysyła do serwera żądanie send_topscores_list i oczekuje na potwierdzenie.

   C: **send_topscores_list** =>S

6. Serwer odbiera żądanie send_topscores_list, po czym wysyła wiadomość o następującej składni:

S: **send_topscores_list** <imie1> <wynik1> …… <imieN> <wynikN> =>C

Gdzie:

<imieN> - imię gracza o N-tym wyniku,
<wynikN> - wynik N-tego gracza.

7. Klient chce pobrać dany poziom. W tym celu wysyła do serwera wiadomość send_level, po czym czeka na dane od serwera.

   C: **send_level** level_N=>S

8. Serwer po otrzymaniu od klienta wiadomości send_level wysyła wszystkie potrzebne informacje w danej kolejności. Klient wiedząc, który parametr odpowiada za co może je odpowiednio przypisać w swoim programie.

   S: **send_level** level_N

   <ilosc wrogow> <czas odświeżania> (…)

9. Po zakończeniu rozgrywki klient wysyła serwerowi wynik gry za pomocą send_player_score:

   C:**send_player_score** <imie> <wynik>=>S

10. Serwer odbiera wiadomość send_player_score. Próbuje zapisać otrzymany wynik do przechowywanego pliku highscores.txt. Jeżeli się to uda, serwer odpowiada wiadomościami send_player_score OK. Jeżeli cokolwiek nie wyjdzie serwer send_player_score ERROR.

    S:**send_player_score** OK => C

    LUB

    S:**send_player_score** ERROR => C

11. Klient postanawia zakończyć łączność sieciową z serwerem. W tym celu wysyła do serwera żądanie logout i oczekuje na odpowiedź.

    C:**logout**=>S

12. Serwer po otrzymaniu żądania logout wysyła wiadomość loggedout, po czym połączenie zostaje zamknięte, a serwer usuwa danego klienta z wektora aktualnie obsługiwanych klientów.

    S:**loggedout**=>C