Karolina Olszewska
Hubert Rutkowski

Protokół sieciowy - aplikacja Space Invaders

Klient:

Po uruchomieniu aplikacji Space Invaders klient powinien zalogować się nawiązując połączenie z serwerem - powinien podać adres hosta i numer portu (alternatywnie klient może wybrać, aby aplikacja działała z lokalnych plików). Po udanym zalogowaniu klient wysyła żądanie do serwera, aby otrzymać dane z plików konfiguracyjnych. Po skończonej grze klient wysyła do serwera żądanie, by jego wynik został wpisany do rankingu wyników.

Serwer:

Stanowi niezależną bezobsługową aplikację. Po uruchomieniu odczytuje dane z plików konfiguracyjnych i na żądanie klienta wysyła te dane do niego. Również na żądanie klienta zapisuje wynik końcowy rozgrywki klienta otrzymany od niego do pliku z rankingiem wyników. Z serwera może jednocześnie korzystać wielu klientów - serwer obsługuje każdego klienta, korzystając z innego wątku.

Protokół:

Jego zadaniem jest nawiązanie połączenia między aplikacją serwera a aplikacją klienta. Umożliwia komunikację i wymianę danych pomiędzy serwerem a klientem - pobieranie danych z plików konfiguracyjnych aplikacji klienta, opisu poszczególnych poziomów gry, rankingu najlepszych wyników. Jest to protokół tekstowy. Wykorzystuje gniazda komunikacyjne i protokół TCP. Pakiet danych jest przesyłany w jednej linii - znak końca linii świadczy o końcu przesyłania danych. Pakiety mogą mieć zatem różny rozmiar i różną liczbę znaków.

Działanie protokołu:

1. Klient wysyła wiadomość do serwera:
   Klient: Wiadomość -> Serwer
2. Następuje nawiązanie połączenia przy pomocy protokołu TCP. Klient wysyła informację do serwera, że chce nawiązać z nim połączenie poprzez wysłanie komunikatu o logowaniu:
   Klient: Login \n -> Serwer
3. a) Serwer otrzymuje komunikat i zaakceptuje połączenie - wysyła odpowiedź do klienta, że ten został zalogowany:
   Serwer: Logged_in \n -> Klient
   b) Serwer otrzymuje komunikat i nie akceptuje połączenia - wysyła odpowiedź do klienta informującą, że próba nawiązania połączenia została odrzucona:
   Serwer: Connection_failed \n -> Klient
4. Jeśli połączenie uda się nawiązać jest możliwa dalsza komunikacja pomiędzy aplikacjami. Klient przesyła do serwera informację o chęci pobrania pliku konfiguracyjnego:

Klient: Get_conf \n -> Serwer

5. Serwer odpowiada klientowi poprzez wysłanie danych zawartych w pliku konfiguracyjnym:
   Serwer: Give_conf: gameHeight gameWidth …\n -> Klient
6. Klient przesyła do serwera informację o chęci pobrania listy z rankingiem najlepszych wyników:
   Klient: Get_High_Scores \n -> Serwer
7. Serwer odpowiada klientowi poprzez wysłanie listy z najlepszymi wynikami:
   Serwer: Give_High_Scores nick1 score 1; … nick10 score10 \n -> Klient
8. Klient przesyła do serwera informację o chęci pobrania dostępnych poziomów gry:
   Klient: Get_levels \n -> Serwer
9. Serwer odpowiada klientowi poprzez wysłanie liczby dostępnych poziomów:
   Serwer: Give_levels: first:1; last:3 \n -> Klient
10. Klient przesyła do serwera informację o chęci pobrania pliku konfiguracyjnego konkretnego poziomu (x):
    Klient: Get_level: x \n -> Serwer
11. Serwer odpowiada klientowi poprzez wysłanie danych danego poziomu:
    Serwer: enemyNumber enemyColor colorBackground… \n -> Klient
12. Klient przesyła do serwera informację o uzyskanym wyniku poprzez wysłanie swojego nicku i wartości wyniku:
    Klient: Score: nick score \n -> Serwer
13. Serwer sprawdza, czy wysłany wynik zostanie umieszczony w rankingu 10 najlepszych wyników.
    Wynik zostanie umieszczony. Serwer wysyła do klienta informację:
    Serwer: Score_attached \n -> Klient
    Wynik nie zostanie umieszczony. Serwer wysyła do klienta informację:
    Serwer: Score_rejected \n -> Klient
14. Klient przesyła do serwera informację o chęci zakończenia połączenia:
    Klient: Log_out \n -> Serwer
15. Serwer otrzymując taką informację, wysyła do klienta komunikat mówiący o zakończeniu połączenia i zamyka to połączenie:
    Serwer: Logged_out \n -> Klient
    Po otrzymaniu wiadomości klient kończy połączenie.
16. W przypadku awarii/wystąpienia błędu po stronie serwera do klienta jest wysyłany komunikat o awaryjnym zamknięciu serwera:
    Serwer: Lose_connection \n -> Klient
17. Klient otrzymuje komunikat, odpowiada o zakończeniu połączenia i zamyka to połączenie z serwerem:
    Klient: Lost_connection \n -> Serwer
18. Po otrzymaniu komunikatu od klienta serwer zamyka połączenie z klientem.