

## Serwer Lunar Lander - etap IV

### Działanie serwera:

- Serwer jest osobną, samodzielną aplikacją.
- W plikach lokalnych przechowuje pliki konfiguracyjne oraz ranking.
- Serwer pobiera numer portu
- Po włączeniu serwer pokazuje numer portu oraz adres IP z którymi może połączyć się klient.
- Po udanym zalogowaniu serwer wysyła do klienta plik konfiguracyjny.
- Serwer posiada konsole z listą w której umieszczane są prośby klienta i odpowiedzi serwera. (Ten punkt uda się jeśli nie będzie to wpływało na szybkość aplikacji serwera)
- Po każdorazowym rozpoczęciu poziomu serwer wysyła na żądanie klienta wiadomość, że istnieje. Od razu po tym na żądanie klienta wysyła mu plik konfiguracyjny z właściwościami danej mapy.
- Serwer na żądanie klienta zapisuje wynik gry i nick gracza do rankingu sieciowego, jak również wysyła mu cały ranking.

### Działanie klienta:

- Po uruchomieniu klienta użytkownik ma możliwość gry offline lub online, czyli z połączeniem z serwerem. Aby grać online należy wpisać adres IP oraz podać numer portu w odpowiednich polach w oknie.
- Po udanym zalogowaniu klient pobiera z serwera pliki konfiguracyjne, w przypadku nieudanego logowania klient otrzyma informacje o niepowodzeniu logowania (serwer offline lub nieprawidłowe dane do podłączenia)
- Przy rozpoczęciu każdego poziomu klient wysyła żądanie do serwera aby sprawdzić czy serwer jest włączony i można z niego pobierać pliki konfiguracyjne do odpowiedniego poziomu. Jeśli jest niedostępny gra przechodzi w tryb offline.
- Po pomyślnym lub nieudanym zakończeniu gry klient znów wysyła żądanie do serwera aby sprawdzić czy serwer istnieje, po czym wysyła żądanie aby zapisać wynik gracza w rankingu sieciowym.
- Gracz może obejrzeć ranking sieciowy z pozycji w głównym menu. Po naciśnięciu przycisku program sprawdza czy serwer istnieje wysyłając żądanie, jeśli istnieje wysyła żądanie aby pobrać listę wyników.

### Protokół:

- Linia tekstu kończąca się znakiem nowej linii, typ tekstowy

## **Działanie protokołu:**

Gdy klient wysyła żądanie do serwera za każdym razem ustanawiane jest nowe połączenie, po otrzymaniu odpowiedzi na żądanie połączenie zostanie przerwane.

### **Client: „GetConfigs” => Server**

Klient prosi serwer o plik konfiguracyjny zawierający ilość żyć statku, prędkości statku itp. Serwer wysyła te dane w postaci obiektu String.

### **Client: "GetRanking" => Server**

Klient prosi serwer o plik zawierający listę wyników graczy. Serwer wysyła tę listę w postaci obiektu String.

### **Client: "GetLevel-" + levelNumber => Server**

Klient prosi serwer o plik konfiguracyjny zawierający właściwości poziomu o podanym numerze. Serwer wysyła te dane w postaci obiektu String.

### **Client: "Check" => Server**

Klient prosi serwer o wiadomość czy serwer istnieje. Serwer wysyła wiadomość „connected”, jeśli nie wyśle nic, lub cokolwiek innego aplikacja przechodzi w tryb offline.

### **Client: ("SaveScore" + "-" + nick + "-" + points => Server**

Klient prosi serwer o zapisanie wyniku gracza w pliku Ranking.txt. Serwer wysyła wiadomość jeśli zostało pomyślnie zapisane.

Serwer wstępnie działa, klient pomyślnie się z nim łączy. Do dopracowania została nam konsola i ewentualna szybkość całego serwera.