**AdiTennis protokół Serwer – Klient**

**dokumentacja. v1.0**

**NAWIĄZANIE POŁĄCZENIA**

|  |  |
| --- | --- |
| **SEWER** | **KLIENT** |
| BROADCAST Odpowiada strukturą z danimi  [ProtoContract]  class ServerInfoStruct  {  [ProtoMember(1)] public string StructId =  "ADITENNIS\_SERVERINFOSTRUCT";  [ProtoMember(2)] public int Port;  [ProtoMember(3)] public string PlayerNick;  [ProtoMember(4)] public string IpAddress;  }  TCP Oczekuje na nick, nie dłużej niż 3 sekundy od połączenia  Odpowiada nr odpowiedzi.  [ProtoContract]  public enum ServerToClientResponseEnum  {  ~~None = 0,~~  ConnectionAccepted = 1,  ConnectionRefused = 2,  ConnectionError = 3,  BusyServer = 4,  ChangeNick = 5  }  TCP  W przypadku innej odpowiedzi niż:  ConnectionAccepted  Połączenie jest zrywane przez serwer. | BROADCAST  Nadaje sygnał DISCOVERY  "ADITENNIS\_DISCOVERY"  TCP  WYSYŁA SWÓJ NICK  "JESTEM\_KLEINTEM" |

**ROZGRYWKA (TCP)**

**Każdy tym wiadomości może zostać wysłany niezależnie.**

**Nie obowiązuje żadna kolejność.**

|  |  |
| --- | --- |
| **SEWER** | **KLIENT** |
| Dokonuje wszelkich obliczeń. Po każdym przeliczeniu wysyła aktualny stan gry. Stan gry to klasa „TennisState” na którą składają się:  [ProtoContract]  class TennisState  {  [ProtoMember(1)] public volatile int BallPosX;  [ProtoMember(2)] public volatile int BallPosY;  [ProtoMember(3)] public string[] PlayerNickname;  [ProtoMember(4)] public volatile int[] PlayerPoints;  [ProtoMember(5)] public volatile int[] RacketPosY;  [ProtoMember(6)] public volatile bool GameOver;  [ProtoMember(7)] public volatile PlayerTypeEnum WhoWon;  }  enum PlayerTypeEnum  {  Server = 0,  Client = 1  }   * Współrzędne piłki * Nicki graczy * Aktualny stan punktowy * Aktualne położenie rakietek * Informacja, czy gra się zakończyła \* * Informacja który gracz wygrał \*   \* - informacja zostaje dostarczona w ostatnim statusie. | Wysyła każde naciśniecie przycisku –  strzałka w górę, lub w dół.  Obiekt typu ConsoleKeyInfo |