Remote Snake API

May 10, 2019

1 General Description

- All transmissions will be based on TCP because:
  - Packet length are not fixed
  - Packet ordering is important
- All TCP stream from **client to server** will:
  - Contain plain json data
  - Be terminated by a "#EOF" line (in order for the server to detect the end of the request
- All TCP stream from **server to client** will contains plai json data (connection will be closed by the server so there is no need of "#EOF".

2 Communications

2.1 Initialisation

1. Server wait for a client
2. Client can send:

   ```
   {
       "type": "new-game"
   }
   #EOF
   ```

3. Server can reply:

   ```
   {
       "type": "state",
       "game-id": 1,
       "game-over": false,
       "snake": [[1,2],[1,3]],
       "food": [[6,7]]
   }
   ```

2.2 Gameplay

2.2.1 Change Direction

1. When client is playing a game it can ask to the server to change snake direction:

```
{
    "type": "update",
    "game-id": 1,
    "direction": "left",
}
#EOF
```

2. Server can reply

```
{
    "type": "state",
    "game-id": 1,
    "game-over": false,
    "snake": [[0,2],[1,2]],
    "food": [[6,7]]
}
```

2.2.2 Refresh Screen

1. When no key are press (the snake is simply going straigth forward). So, client can send:

```
{
    "type": "update",
    "game-id": 1,
    "direction": null
}
#EOF
```

2. Server can reply:

```
{
    "type": "state",
    "game-id": 1,
    "game-over": false,
    "snake": [[1,2],[0,2]],
    "food": [[6,7]]
}
```

2.2.3 End Game

- When game is over server will send the following state message (switch game-over to true):

```
{
    "type": "state",
    "game-id": 1,
    "game-over": true,
    "snake": [[0,2],[1,2]],
    "food": [[6,7]]
}
```

- No reply is expected from the client and server will be in charge to free local memory.