NSPong

Documentation
v1.0

Tuomas Seppänen
Sampsa Sarjanoja
Sakari Alapuranen

Contents

NSPong documentation

NSPong is a HTML5 demo game developed on top of ARM Sensinode's NanoService Platform. The game uses for example RealTimeMultiplayerNodeJS, Box2D and CAAT libraries, which are specifically built for HTML5 multiplayer games with the client/server model.

1. Installation instructions

Steps to install NSPong:

1. Download NSPong repository from https://github.com/NSPong/NSPong
2. Download NS_Game_Controller software from http://mbed.org/teams/NSPong/code/NS_Game_Controller/ and install it to your mbed devices
3. Download and install node.js from http://nodejs.org
4. Download and unzip NanoService Platform from http://silver.arm.com
5. Run NanoService Platform from /nsp-devel/bin/runNSP.bat
6. Open terminal and navigate to the root directory of this repository
7. Run "npm install"
8. Run "node js/NSPong/server.js"
9. Open browser and open http://localhost:4004

2. How to play

2.1. Game description

Connect two mbed application boards with LPC1768 modules with different IP addresses to the same NSP host. Make sure that they both can find and register to the NSP host by visiting https://<nsp-host-address>:8081 and checking that they are listed in the end-points listing view. The default credentials are admin:admin. When the game server is running, the boards should beep and show a text on their screens when they are registered to the NSP. After successful registrations game can be started by first selecting the left player by clicking the joystick on the board and then selecting the right player by clicking the joystick on another board. After second board has appeared on the game screen the ball starts to move by itself and the game is on. The idea is to not let the ball touch the wall behind the paddle you are controlling. If the ball touches the wall colored by red, the opposite player gets a point. The game starts over when a player reaches five points.

The game can be reset by clicking the game area with mouse. When the game has been reset, the players can join back and start a new game by clicking the joystick on the board in the order they want.

2.2. Board interaction

A board, which is connected to the game, outputs accelerometer data of two axes (X and Y) to the game server rapidly, which is used to control the paddles in the game. Controlling the paddle consists of moving the board across the game area and tilting the board. When a player joins the game by clicking the joystick, the text on the LCD screen changes accordingly. When the ball hits a paddle, the buzzer on the corresponding board beeps and the RGB led flashes white. After a goal, a sound is buzzed on the scoring player's board and the RGB led rotates multiple colors. The four blue LEDs on the boards show the score of the player. When a player has won the game a victory sequence is played on the board.

3. Demo video

A video of the gameplay is available at: https://vimeo.com/95207889

4. Implementation

4.1. Game server

The game server is implemented in JavaScript and is run by using Node.js interpreter. The game server code is based on a real-time multiplayer game engine developed by Mario Gonzalez. The server uses several 3rd party libraries:

- RealTimeMultiplayerNodeJS - Base for developing a real-time multi-client HTML5 game
- Node.js - An interpreter for running JavaScript on the server side
- Socket.IO - Provides client-server communication via WebSockets, etc.
- Express - Library for creating HTTP servers easily
- Box2DJS - Library for realistic 2D physics
- CAAT - For rendering graphics on the client side

The visuals are done by utilizing new CSS3 techniques, which include drop-shadows for example. The game rendering uses CAAT library for creating simple shapes on the canvas.

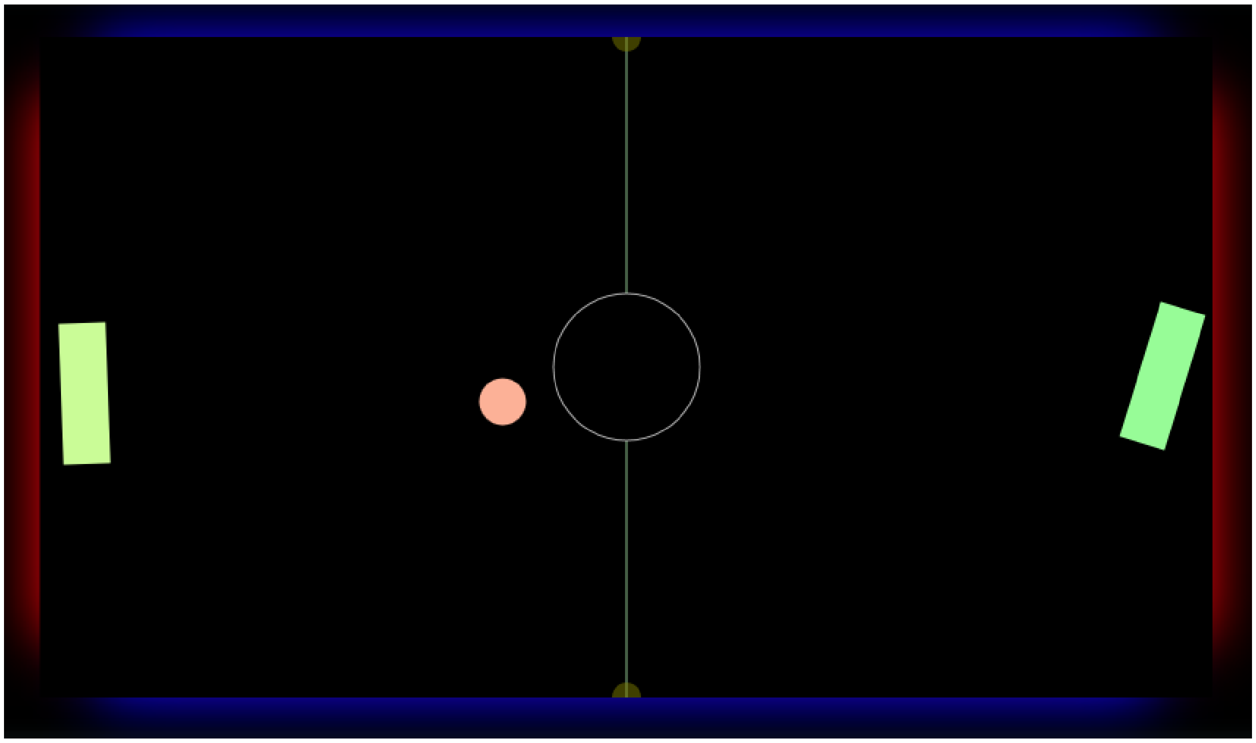
Image 1. NSPong gameplay

4.2. mbed software

The mbed board controller software is based on the template NSP application code available by ARM. New resources have been added and some changes: faster sample rate setting for accessing accelerometer data, network address in use and removal of interval registration to NSP.

4.2.1. Resources

The mbed boards have the following resources:

- Accelerometer
  - By subscribing to the accelerometer resource the web server is able to get the values of the chosen accelerometer axes
  - Web server can also change the axes to the ones it wants to observe

- LED
  - LED resource consists of four blue LEDs and one RGB LED
    - Blue LEDs are used to indicate the score of the player
      - When a player wins the game, all the blue LEDs flash
    - RGB LED is used to show an effect when a player has scored
    - RGB LED also flashes when the ball hits a paddle

- LCD
    - LCD displays information about the user's registration to the web server
    - After the user has registered and pressed the joystick, the LCD shows a message telling the user the position on the game board

- Buzzer
    - There are three different buzzer sequences.
        - When the player's paddle is hit, a single beep sequence is played
        - When the player scores, a score sequence is played
        - When the player wins the game, a victory sequence is played

- Joystick
    - Joystick is used to register to the game.
        - After the player sees the LCD registration text, the player can press the joystick to register to the game

4.2.2. REST API documentation

REST calls are made to the NanoService platform, which forwards them to the board as CoAP messages. Default port in use is 8080.

Example: HTTP PUT http://localhost:8080/domain/endpoints/<endpointname>/<resource>

<body>

All REST calls require HTTP basic authorization. The default username and password are *admin* and *secret.* The resources introduced above have the following REST methods:

- Accelerometer
    - PUT
        - Choosing the axes of which the board will send accelerometer data
        - Headers
            - Content-type: text/plain
        - Body
            - Selected combination of accelerometer axes
            - "xyz", "xy", "xz", "yz", "x", "y", or "z" allowed
    - GET
        - Fetching the accelerometer value(s) for selected axes

- LED
    - PUT
        - Lights up LEDs on the board based on the message in body
        - Headers

            - Content-type: text/plain
        - Body
            - “reset”, “score”, “paddle” or “win” allowed

- LCD
    - PUT
        - Displays text on screen based on the message in body
        - Headers
            - Content-type: text/plain
        - Body
            - “1”, “2” or “info” allowed

- Buzzer
    - PUT
        - Sounds the buzzer based on the message in body
        - Headers
            - Content-type: text/plain
        - Body
            - “beep”, “score” or “win” allowed

- Joystick
    - GET
        - Fetching the position of the joystick