Pong exercises

Exercise: Make changes to the Pong game

We have a simple Pong game programmed in TypeScript.

In this exercise you must:

1. Read and understand other programmers code (the Pong game)
2. Extend the Pong game.

Getting ready

1. Clone the Pong game  
   git clone [https://github.com/dimselab/pong](https://www.google.com/url?q=https://github.com/dimselab/pong&sa=D&ust=1570110076428000)
2. Download the necessary modules  
   npm install
3. Open Visual Studio Code  
   code .
4. Run the WebPack watcher  
   npm run watch
5. Now the Pong game should run in your browser.  
   You are allowed to try the game …  
   If you don’t know how to control the player look in the code ...

Read the code

Read the TypeScript code of the Pong game.

Noticeable program elements (and some questions … write the answer as comments in the program)

* The game is structured in more files  
  Why do you think it is like that?
* Canvas element in HTML  
  What is the purpose of the canvas element?
* Import and export keywords  
  What is the semantics (aka meaning) of import and export
* Interface and class  
  What is the difference between interfaces and classes?
* The class Vector  
  What is the purpose of this class?
* The game loop  
  Where is the loop actually?
* Framerate  
  What does frame rate mean?

Reduction: Remove the frame rate

The frame rate should no longer be shown on the screen.

Change the colors

Change the background color of the canvas element.

Change the color of the player and the ball.

Extension: Stop the game

The game must stop if the ball is behind the player, that is to the left of the player.

Hint: Can be done in the game loop.

The game can be restarted by reloading the web application in the browser.

Extension: Count points

Every time the player hits the ball (s)he earns a point.

The points must be updated and shown on the screen.

You should add another div element (outside the canvas)  to show the points.

Change the size of the canvas

Make the *canvas*element bigger.

Harder extension: Another player

Add a Player2 to the game. Player2 must be positioned to the right of the canvas.

Player2 is controlled by the “o” and “l” keys (requires some refactoring to the Player class)

Player2 is dead if the ball is behind (to the right of) Player2.

Count points for each player: Add another div element for Player2 points.