Documentation

In order to show case the learnings in HTML, CSS and JavaScript, a Game website was created.

This document will cover the game structure and code

On the initial page called *home.html* , the standard structure as the website was kept.

The interface was developed in the file *style1.css.*

Accessing the tab “game” we can visualize the structure created on *game.html*, and contains, besides the main bar, our canvas tab.

```
<body onload="startGame()">
<main>
        <h3>Ping Pong</h3>
        <p>Move the mouse to operate the left paddle and catch the  balls</p>
        <br>

        <canvas id="gameCanvas" width="800" height="600"></canvas>

        </main>
```

The game is created on the file *1main.js.*

All the code is commented, with explanations about the logical procedure (step by step)

I started defining the variables and worked initially with the x achse.

```
var canvas;
var canvasContext;
var ballX = 50;
var ballY = 50;
var ballSpeedX = 10;
```

After work only the the x achses, I started to give the movement of the ball also in the vertical way. (check moveEverything)

```
var ballSpeedY = 4;

var ballSize = 10;

var audioBall = new Audio('sound/shot.mp3');

var audioWall = new Audio('sound/spawn.mp3');

var audioWin = new Audio('sound/winner.mp3');


var player1Score = 0;

var player2Score = 0;

const WINNING_SCORE = 3;//maximum value of score.

var showingWinScreen = false;

var paddle1Y = 250;//Left

var paddle2Y = 250;//right

const PADDLE_THICKNESS = 10;

const PADDLE_HEIGHT = 100;
```

The idea now is to build the paddles and make the screen recognize where our mouse is. Then, our paddle needs to follow our information. First we build the calculateMousePos, wich answers an event. Then, on the window.onload we need to add a canvas.addEventListener for our mousePos(check it). And the last step, is to put the var 'paddle1Y' on the function 'drawEverything', to make the draw recognize all the code and calculations we made.

```
function calculateMousePos(evt) {

    var rect = canvas.getBoundingClientRect();

    var root = document.documentElement;

    var mouseX = evt.clientX - rect.left - root.scrollLeft;

    var mouseY = evt.clientY - rect.top - root.scrollTop;
```

```
	return {
		x: mouseX,
		y: mouseY
	};
}

function handleMouseClick(evt) {
	if (showingWinScreen) {
		player1Score = 0;
		player2Score = 0;
		showingWinScreen = false;
	}
}
window.onload = function () {
	canvas = document.getElementById('gameCanvas');
	canvasContext = canvas.getContext('2d');
	canvasContext.font = "30px Arial";
```

We now are going to set an interval for drawEverything. What we want is that the red retangle moves. Once we defined our 'var ballX', we put it in the function drawEverything the ballX+20, so the ball walks 20 pixels to the right. The red retangle defined with fillRect, recieved the "var ballX" in the place of any number, so then we make it move.

```
	var framesPerSecond = 30;
	//setInterval(drawEverything, 1000/framesPerSecond);as the draw evoluted, was needed a function to call draw and move at the same time.
	setInterval(function () {
		moveEverything();
		drawEverything();
	}, 1000 / framesPerSecond);
```

```
	canvas.addEventListener('mousedown', handleMouseClick);//for the 'click to continue'

	canvas.addEventListener('mousemove', function (evt) {

		var mousePos = calculateMousePos(evt);

		paddle1Y = mousePos.y - (PADDLE_HEIGHT / 2);


	 //y was defined on the 'return' in 'calculateMousePos'. if I didnt define this position(paddle_height/2), my mouse was far from the paddle. This way my mouse is on the bottom of my paddle, wich guarantees more control of the paddle and allows me to get in the very top and very bottom of the screen.

	});

}


function ballReset() {
```

The idea of this function is to make the ball restart when the paddle does not catch it on the left side. The re-start will be start from the center of the screen. So we make this function and apply it to the moveEverything = where we have the x and the left side. In other words, if(ballX<0)... check it out in the function.

Then, we have, “IF” the ball is re.starting is because somebody scored something, so we are going to apply the score reset here too.

```
	if (player1Score >= WINNING_SCORE ||

		player2Score >= WINNING_SCORE) {

		showingWinScreen = true;

	}

	if (player1Score >= WINNING_SCORE) {

		audioWin.play(); //audio

	}

	ballSpeedX = -ballSpeedX;

	ballX = canvas.width / 2;

	ballY = canvas.height / 2;

}
```

```
function computerMovement() {//for the paddle2Y
	var paddle2YCenter = paddle2Y + (PADDLE_HEIGHT / 2);
	if (paddle2YCenter < ballY+35) {
//var paddle2YCenter so the ball will be catched in the center of my paddle.
		paddle2Y += 6;// same as say: paddle2Y = paddle2Y+ 6;
	} else if (paddle2YCenter > ballY-35) {
//35 is a random number to make the paddle stop shaking.
		paddle2Y -= 6;
	}
//this code means: if the ball is above the paddle, the paddle catches in +6, if its under, the paddles moves -6.
}

function moveEverything() {
	if (showingWinScreen) {
		return;
	}
	computerMovement();
	ballX += ballSpeedX; //ballX = ballX + ballSpeedX;
	ballY += ballSpeedY;

	/*############Determing the ball going to the left and right###############*/

	if (ballX < 0) {// making the same on the left side. ball needs to touch the wall and go to the right.
		//ballSpeedX = -ballSpeedX --> this was cutted and added to the function ballreset, wich will restar the ball from the center and flip it to the opposite side.
```

```
            //ballReset();

            //lets add the fact that if the ball touches on the paddle, it goes in the opposite direction. Otherwise, ballreset.

            if (ballY > paddle1Y &&

                    ballY < paddle1Y + PADDLE_HEIGHT) {

                    ballSpeedX = -ballSpeedX;

                    audioBall.play(); //audio

                    //

                    var deltaY = ballY -

                            (paddle1Y + PADDLE_HEIGHT / 2);//where the ball hits + middle of the paddle

                    ballSpeedY = deltaY * 0.35;

                    //random number to slow the speed down. This makes the ball go slower when hits the center of the paddle. When hits an edge and makes an angle, speeds up the ball movement.

            } else {

                    audioWall.play();

                    player2Score++; // must be BEFORE ballReset()

                    //in other words: player2Score+=1

                    ballReset();

            }

    }




    if (ballX > canvas.width) {

             // we want the ball to touch the wall and go back. Therefore we could use ballX>800, but if we make the canvas bigger or smaller, this code wouldn't work properly. So the best way is to take off the number and refer to the canvas.width.
```

```
            //solution 1:'ballSpeed=-5;'in case we change the number of the ballSpeed we have two different speeds: one left to right and one right to left. So the best way to write this sentence is:

            //                  ballSpeedX = -ballSpeedX

            //now, lets make the right paddle answers the ball, like the paddle1

            if (ballY > paddle2Y &&

                        ballY < paddle2Y + PADDLE_HEIGHT) {

                        ballSpeedX = -ballSpeedX;

                        audioBall.play(); //audio

                        //

                        var deltaY = ballY -

                                    (paddle2Y + PADDLE_HEIGHT / 2);//where the ball hits + middle of the paddle

                        ballSpeedY = deltaY * 0.35;//random number to slow the speed down. Wich means, when the ball hits the center of the paddle, the ball go slowlier and horizontal position. we have bigger speed when te ball hits the edge of the padle and increases the angle of the ball too.

            } else {

                        audioWall.play();

                        player1Score++; // must be BEFORE ballReset()

                        ballReset();

            }

      }

/* in both directions this code is the same, because:

      ex: if ballspeed is (-5), then, we were gonna have: ballSpeedX= -(-5)*/

      //now, lets give the movement of the ball in the y achse. Just copy paste the ballX and rename.


 //movin ghe ball horizontaly
```

```
	if (ballY < 0) {
		ballSpeedY = -ballSpeedY;
	}
	if (ballY > canvas.height) {
		ballSpeedY = -ballSpeedY;
	}
}

function drawCourt() {//first, the net
	for (var i = 0; i < canvas.height; i += 40) {
		colorRect(canvas.width / 2 - 1, i, 2, 20, 'white');
	}//down here, we have the court lines
	colorRect(0, 60, canvas.width,1,'white');
	colorRect(0, 540, canvas.width,1,'white');

	colorRect(200, 60, 1,480,'white');
	colorRect(600, 60,1, 480,'white');

	colorRect(200,300,400,1,'white')
}

function drawEverything() {
	// next line blanks out th escreen with green.

	colorRect(0, 0, canvas.width, canvas.height, 'green');
```

```
	if (showingWinScreen) {

		if (player1Score >= WINNING_SCORE) {
			canvasContext.fillStyle = '#adff2f';
			canvasContext.fillText("You Won!", 350, 200);
		} else if (player2Score >= WINNING_SCORE) {
			canvasContext.fillStyle = '#adff2f';
			canvasContext.fillText("You didn't win. Try again.", 250, 200);
		}
		canvasContext.fillStyle = '#adff2f';
		canvasContext.fillText("click to continue", 300, 500);
		return;

	}

	drawCourt();
	// this is LEFT player paddle.
	colorRect(0, paddle1Y, PADDLE_THICKNESS, PADDLE_HEIGHT, '#f4187d');
	// this is RIGHT computer paddle.
	colorRect(canvas.width - PADDLE_THICKNESS, paddle2Y, PADDLE_THICKNESS,
PADDLE_HEIGHT, 'orange');// in the other side of the screen we have to discount the thickness
of the paddle (otherwise the paddle wont show on the canvas)

	//colorRect(ballX,100,10,10,'red');

	canvasContext.fillText(player1Score, 100, 100);
```

```
	canvasContext.fillText(player2Score, canvas.width - 100, 100);// minus 100 to give me the same distance in the canvas but in the oposite side of my tennis court.



	// next line draws the ball.

	colorCircle(ballX, ballY, ballSize, 'white');

	/*

		ctx.fillStyle= 'red';// if i draw the white box after the red one, when the red starts to move, the white, that was written after, will overlapse the red. Therefore, we must write the red AFTER the white box.

	ctx.fillRect(ballX,100,10, 10);

	ctx.fillStyle= 'green';

	ctx.fillRect(50,230,10,10);

	ctx.fillStyle= 'blue';

	ctx.fillRect(400,300,40,40);//I could also say (canvas.width/2, canvas.height/2, 40,40)*/




}



function colorCircle(centerX, centerY, radius, drawColor) {

	//colorRect(ballX,100,10,10,'red');

	//making the ball round: took the colorRect from the 'drawEverything function and make this one to make it clear. -->1)how we want it to look:

	//ctx.fillStyle = 'white'

	//ctx.beginPath();

	//ctx.arc(ballX, 100, 10, 0, Math.PI * 2, true);

	//ctx.fill();

	//2. we first are going to put the 'ugly' function for the circle here and the 'pretty' one is going to the drawEverything
```

```
	//colorRect(ballX,100,10,10,'red');
	//making the ball round
	canvasContext.fillStyle = drawColor;
	//Achtung! drawColor is without the ''.
	canvasContext.beginPath();
	canvasContext.arc(centerX, centerY, radius, 0, Math.PI * 2, true);
	canvasContext.fill();
}

function colorRect(leftX, topY, width, height, drawColor) {
	/*this is a little adapt from the drawEverything.
	Instead using
	ctx.fillStyle= 'black';
	ctx.fillRect(0,0,canvas.width, canvas.height);
	ctx.fillStyle= 'white';
	ctx.fillRect(0,210,10,100); we are going to give new names:*/
	canvasContext.fillStyle = drawColor;
	//instead 'black' we use the parameter that we gave in the function. The color we describe in the function drawEverything
	canvasContext.fillRect(leftX, topY, width, height);
}
```