MOTIVATION

Pong ist eines der ersten und bekanntesten Videospiele, das 1972 von Atari veröffentlicht wurde. Es gilt als eines der ersten kommerziellen Videospiele und hat einen bedeutenden Einfluss auf die Entwicklung der Videospielindustrie.

Es ist ein einfaches Sportspiel konzipiert, das Tischtennis simuliert. Zwei Spieler steuern Schläger (Paddles), die sich vertikal auf der linken und rechten Seite des Bildschirms bewegen. Ziel des Spiels ist es, den Ball (ein Quadrat oder Kreis) zu treffen und ihn so zu spielen, dass der Gegner ihn nicht zurückschlagen kann.

ENTWICKLUNG DES SPIELS

1.) Der Hintergrund soll schwarz, die Schläger (Paddle) und Ball weiß sein. Zudem wird die Position absolut angegeben, um Ball und Schläger schrittweise bewegen zu können.

```
body {
   background-color: black;
   }
```

```
#paddleLeft, #paddleRight, #Ball {
  background-color: white;
  position: fixed;
  }
```

2.) Ball, Paddle werden mit Abkürzungen referenziert.

```
const padLeft = document.getElementById("paddleLeft");
const padRight = document.getElementById("paddleRight");
const ball = document.getElementById("Ball");
```

3.) Die absoluten Positionen von Paddle und Ball werden initialisiert. Das passiert im Script-Teil, da diese Attribute sonst im Script nicht angesprochen werden können.

Die Abkürzungen vw und vh stehen für *Viewport Width* und *Viewport Height.* Diese Einheiten beziehen sich auf die Breite und Höhe des Ansichtsfensters (Viewport) in Prozent und werden in CSS verwendet, um Elemente relativ zur Größe des Viewports zu gestalten. 1 vw entspricht 1% der Breite und 1 vh entspricht 1% der Höhe des Viewports (Fensters).

```
ball.style.top = "50vh";
ball.style.left = "50vw";
ball.style.width = "2vw";
ball.style.height = "2vw";
var rtgX = 3;
var rtgY = 2;
```

```
padLeft.style.top = "50vh";
padLeft.style.left = "2vw";
padLeft.style.height = "20vh";
padLeft.style.width = "2vw";
```

```
padRight.style.top = "50vh";
padRight.style.left = "94vw";
padRight.style.height = "20vh";
padRight.style.width = "2vw";
```

4.) die Spielumgebung (Windos) bekommt einen EventListener, um Tasteneingabe zu verarbeiten. Beim Drücken einer Taste wird der EventListener, der am Window hängt, ausgelöst. Das übergebene Objekt „event“ enthält alle Informationen über das Ereignis, unter anderem auch die gedrückten Tasten. Die numerischen Werte von vh und vw lassen sich über **parseInt(„50vh“) = 50** ermitteln. Damit werden die Paddles um 5% der Bildschirmhöhe bewegt.

```
window.addEventListener("keydown",bewegen);

function bewegen(event) {
 if (event.key === "ArrowUp") {
  padRight.style.top = parseInt(padRight.style.top) – 5 + "vh";
   } else if (event.key === "ArrowDown") {
      padRight.style.top = parseInt(padRight.style.top) + 5 + "vh";
   } else  if (event.key === "a") {
       padLeft.style.top = parseInt(padLeft.style.top) - 5 + "vh";
   } else  if (event.key === "y") {
       padLeft.style.top = parseInt(padLeft.style.top) + 5 + "vh";
   }
  }
}
```

5.) Der „timer“ wird über **setInterval()** gestartet. Um ihn mit **clearInterval()** später wieder anhalten zu können, wird er referenziert.

Referenz = **setInterval**(auszuführende Funktion, ms);
clearInterval (Referenz);

Die Funktionen bewegeBall(), pralleAbRand() und testeStopp() und werden im Timer ausgeführt. Sie bewegen den Ball, lassen ihn oben und unten abprallen und stoppen den Timer, wenn der Ball den linken über rechten Spielfeldrand berührt.

```
var Timer = setInterval(timer, 100);

function timer() {
   bewegeBall();
   pralleAbRand();
   testeStopp();
   }

  function bewegeBall() {
    ball.style.top = parseInt(ball.style.top) + rtgY +"vh";
    ball.style.left = parseInt(ball.style.left) + rtgX +"vw";
    }

function pralleAbRand() {
  if (((parseInt(ball.style.top) < 0)) || parseInt(ball.style.top) >= 96) {
    rtgY = -rtgY;
    ball.style.top = parseInt(ball.style.top) + rtgY +"vh";
    }
}

function testeStopp() {
  if (((parseInt(ball.style.left)<4)) || parseInt(ball.style.left)>=96) {
    clearInterval(Timer);
  }
}
```

6.) Der Abprall des Balls an den Panels wird über eine Kollisionserkennund durchgeführt. Dabei werden jeweils die Ränder der Rechtecke miteinander verglichen. Die Funktion timer wird erweitert. Dabei haben:

Arithmetische Operatoren (+, -, *, /):	höhere Priorität
Vergleichsoperatoren (>, <, >=, <=):	niedrigere Priorität
Logische Operatoren (&&, ||):	noch niedrigere Priorität

```
function timer() {
      bewegeBall();
      pralleAbRand();
      testeKollision(ball, padRight);
      testeKollision(ball, padLeft);
      testeStopp();
      }
```

```
function testeKollision(inA, inB) {
  if (parseInt(inA.style.left)+parseInt(inA.style.width) > parseInt(inB.style.left)
      && parseInt(inA.style.left) < parseInt(inB.style.left) + parseInt(inB.style.width)
      && parseInt(inA.style.top) < parseInt(inB.style.top) + parseInt(inB.style.height)
      && parseInt(inA.style.top) + parseInt(inA.style.height) > parseInt(inB.style.top)) {
          rtgX = (-1)*rtgX;  // Richtungsänderung
         // ein Schritt aus dem Wirkungskreis des Paddles
          ball.style.left = parseInt(ball.style.left) + 3* rtgX +"vw";
      }
   }
```

E R W E I T E R U N G E N

Erweiterung 1
Das Spiel soll über die Space-Taste neu gestartet werden, wenn der Ball im Aus war.
Die Space-Taste wird über *(event.key == " ")* abgefragt.

Erweiterung 2
Das Spiel soll eine Spielstandsanzeige besitzen. Die Anzeige soll den Punktestand des linken Spielers in der linken Hälfte, und den Punktestand des rechten Spielers in der rechten Hälfte zeigen.

Erweiterung 3
Ergänze die gestrichelte Mittellinie.

Erweiterung 4
Nach 10 Ballberührungen soll der Ball schneller werden. Timer dazu erst stoppen und dann neu starten, sonst überlagern sich zwei Timerfunktionen.

Erweiterung 5
Der Ball soll bei einem Abprall zufälliger Weise etwas seinen Abprallwinkel ändern.