**memory game. js**

const gameContainer = document.getElementById("game");

const COLORS = [

"red",

"blue",

"green",

"orange",

"purple",

"red",

"blue",

"green",

"orange",

"purple"

];

// here is a helper function to shuffle an array

// it returns the same array with values shuffled

// it is based on an algorithm called Fisher Yates if you want ot research more

function shuffle(array) {

let counter = array.length;

// While there are elements in the array

while (counter > 0) {

// Pick a random index

let index = Math.floor(Math.random() \* counter);

// Decrease counter by 1

counter--;

// And swap the last element with it

let temp = array[counter];

array[counter] = array[index];

array[index] = temp;

}

return array;

}

let shuffledColors = shuffle(COLORS);

// this function loops over the array of colors

// it creates a new div and gives it a class with the value of the color

// it also adds an event listener for a click for each card

function createDivsForColors(colorArray) {

for (let color of colorArray) {

// create a new div

const newDiv = document.createElement("div");

// give it a class attribute for the value we are looping over

newDiv.classList.add(color);

// call a function handleCardClick when a div is clicked on

newDiv.addEventListener("click", handleCardClick);

// append the div to the element with an id of game

gameContainer.append(newDiv);

}

}

// TODO: Implement this function!

function handleCardClick(event) {

// you can use event.target to see which element was clicked

console.log("you just clicked", event.target);

}

// when the DOM loads

createDivsForColors(shuffledColors);

**memory game. css**

#game div {

border: 1px solid black;

width: 15%;

height: 200px;

margin: 10px;

display: inline-block;

}

**Memory game . js (FANCY SOLUTION)**

document.addEventListener("DOMContentLoaded", function() {

const cards = document.querySelectorAll(".game-card");

let numCards = cards.length;

let card1 = null;

let card2 = null;

let cardsFlipped = 0;

let currentScore = 0;

let lowScore = localStorage.getItem("low-score");

let start = document.getElementById("start");

if (lowScore) {

document.getElementById("best-score").innerText = lowScore;

}

for (let card of cards) {

card.addEventListener("click", handleCardClick);

}

let startBtn = document.getElementById("start-button");

startBtn.addEventListener("click", startGame);

function handleCardClick(e) {

if (!e.target.classList.contains("front")) return;

let currentCard = e.target.parentElement;

if (!card1 || !card2) {

if (!currentCard.classList.contains("flipped")) {

setScore(currentScore + 1);

}

currentCard.classList.add("flipped");

card1 = card1 || currentCard;

card2 = currentCard === card1 ? null : currentCard;

}

if (card1 && card2) {

let gif1 = card1.children[1].children[0].src;

let gif2 = card2.children[1].children[0].src;

if (gif1 === gif2) {

cardsFlipped += 2;

card1.removeEventListener("click", handleCardClick);

card2.removeEventListener("click", handleCardClick);

card1 = null;

card2 = null;

} else {

setTimeout(function() {

card1.classList.remove("flipped");

card2.classList.remove("flipped");

card1 = null;

card2 = null;

}, 1000);

}

}

if (cardsFlipped === numCards) endGame();

}

function startGame() {

setScore(0);

start.classList.add("playing");

let indices = [];

for (let i = 1; i <= numCards / 2; i++) {

indices.push(i.toString());

}

let pairs = shuffle(indices.concat(indices));

for (let i = 0; i < cards.length; i++) {

let path = "gifs/" + pairs[i] + ".gif";

cards[i].children[1].children[0].src = path;

}

}

function shuffle(array) {

let arrayCopy = array.slice();

for (let idx1 = arrayCopy.length - 1; idx1 > 0; idx1--) {

// generate a random index between 0 and idx1 (inclusive)

let idx2 = Math.floor(Math.random() \* (idx1 + 1));

// swap elements at idx1 and idx2

let temp = arrayCopy[idx1];

arrayCopy[idx1] = arrayCopy[idx2];

arrayCopy[idx2] = temp;

}

return arrayCopy;

}

function setScore(newScore) {

currentScore = newScore;

document.getElementById("current-score").innerText = currentScore;

}

function endGame() {

let end = document.getElementById("end");

let scoreHeader = end.children[1];

scoreHeader.innerText = "Your score: " + currentScore;

let lowScore = +localStorage.getItem("low-score") || Infinity;

if (currentScore < lowScore) {

scoreHeader.innerText += " - NEW BEST SCORE!!";

localStorage.setItem("low-score", currentScore);

}

document.getElementById("end").classList.add("game-over");

}

});

**Memory game.css (FANCY)**

/\*

intro to 3d rotations:

https://desandro.github.io/3dtransforms/docs/card-flip.html

\*/

body {

background-color: #c8efff;

}

.btn:hover {

cursor: pointer;

}

#start {

height: 90vh;

font-family: 'Bungee', cursive;

color: #ce29f7;

position: relative;

}

#title {

font-size: 8rem;

perspective: 600px;

}

#title > p {

transform: rotateX( 45deg );

line-height: 0.75;

text-align: center;

color: #610079;

}

#start.playing {

margin-top: -90vh;

transition: margin-top 1s;

}

#start.playing + #game {

display: flex;

}

#game {

display: none;

}

#game .row {

width: 100%;

}

.container {

flex-wrap: wrap;

}

.game-card-wrapper {

perspective: 600px;

width: 75%;

padding-bottom: 75%;

position: relative;

margin: 1rem auto;

}

.game-card-wrapper:hover {

cursor: pointer;

}

.game-card {

width: 100%;

height: 100%;

position: absolute;

transform-style: preserve-3d;

transition: transform 1s;

}

.game-card.flipped {

transform: rotateY( 180deg );

}

.front,

.back {

display: flex;

justify-content: center;

align-items: center;

position: absolute;

backface-visibility: hidden;

width: 100%;

height: 100%;

border-radius: 8px;

border: 2px solid black;

overflow: hidden;

}

.front {

background: linear-gradient(315deg, #ce29f7, white);

font-size: 3rem;

}

.back {

transform: rotateY( 180deg );

}

.back > img {

height: 100%;

width: auto;

}

#current-score {

display: flex;

justify-content: center;

align-items: center;

font-size: 4rem;

font-family: 'Bungee', cursive;

color: #ce29f7;

position: absolute;

top: 0;

right: 0;

bottom: 0;

left: 0;

}

#end {

display: none;

}

#end.game-over {

display: flex;

position: absolute;

background-color: rgba(0,0,0,0.75);

top: 0;

bottom: 0;

left: 0;

right: 0;

flex-direction: column;

align-items: center;

color: white;

justify-content: space-around;

font-family: 'Bungee', cursive;

text-align: center;

}

#end h1 {

font-size: 8rem;

}

#end h4 {

font-size: 3rem;

}

@media (max-width: 992px) {

.game-card-wrapper {

width: 100%;

padding-bottom: 100%;

}

}

@media (max-width: 992px) {

.game-card-wrapper {

width: 120%;

padding-bottom: 120%;

}

}

**Memory game.js (SIMPLE SOLUTION)**

const gameContainer = document.getElementById("game");

let card1 = null;

let card2 = null;

let cardsFlipped = 0;

let noClicking = false;

const COLORS = [

"red",

"blue",

"green",

"orange",

"purple",

"red",

"blue",

"green",

"orange",

"purple"

];

function shuffle(array) {

let counter = array.length;

// While there are elements in the array

while (counter > 0) {

// Pick a random index

let index = Math.floor(Math.random() \* counter);

// Decrease counter by 1

counter--;

// And swap the last element with it

let temp = array[counter];

array[counter] = array[index];

array[index] = temp;

}

return array;

}

let shuffledColors = shuffle(COLORS);

// this function loops over the array of colors

// it creates a new div and gives it a class with the value of the color

// it also adds an event listener for a click for each card

function createDivsForColors(colorArray) {

for (let color of colorArray) {

const newDiv = document.createElement("div");

newDiv.classList.add(color);

newDiv.addEventListener("click", handleCardClick);

gameContainer.append(newDiv);

}

}

function handleCardClick(e) {

if (noClicking) return;

if (e.target.classList.contains("flipped")) return;

let currentCard = e.target;

currentCard.style.backgroundColor = currentCard.classList[0];

if (!card1 || !card2) {

currentCard.classList.add("flipped");

card1 = card1 || currentCard;

card2 = currentCard === card1 ? null : currentCard;

}

if (card1 && card2) {

noClicking = true;

// debugger

let gif1 = card1.className;

let gif2 = card2.className;

if (gif1 === gif2) {

cardsFlipped += 2;

card1.removeEventListener("click", handleCardClick);

card2.removeEventListener("click", handleCardClick);

card1 = null;

card2 = null;

noClicking = false;

} else {

setTimeout(function() {

card1.style.backgroundColor = "";

card2.style.backgroundColor = "";

card1.classList.remove("flipped");

card2.classList.remove("flipped");

card1 = null;

card2 = null;

noClicking = false;

}, 1000);

}

}

if (cardsFlipped === COLORS.length) alert("game over!");

}

createDivsForColors(shuffledColors);

**Memory game.css (SIMPLE SOLUTION)**

#game div {

border: 1px solid black;

width: 15%;

height: 200px;

margin: 10px;

display: inline-block;

}