UDACITY

PROJECT

Memory Game

A part of the Front-End Web Developer Nanodegree Program

PROJECT REVIEW

CODE REVIEW 5

NOTES

▾ js/app.js 4

```
/*
 * Create a list that holds all of your cards
 */
var symbols = ["diamond","diamond","paper-plane-o","paper-plane-o","anchor","anchor","bolt","bolt","cube","cube","leaf","leaf","bi
var open = [];
var moves = 0;
var match = 0;
var stars = 3;
var startTime;

// timer
var timer = setInterval(function() {
    let now = new Date().getTime();
```

AWESOME

Awesome job using the recent javascript keywords (`const` and `let`) to declare variables.

```
    let distance = now - startTime;
    let days = Math.floor(distance / (1000 * 60 * 60 * 24));
    let hours = Math.floor((distance % (1000 * 60 * 60 * 24)) / (1000 * 60 * 60));
    let minutes = Math.floor((distance % (1000 * 60 * 60)) / (1000 * 60));
    let seconds = Math.floor((distance % (1000 * 60)) / 1000);
    let timeString = "";
    if (days > 0) timeString += days + "d ";
    if (hours > 0) timeString += hours + "h ";
    if (minutes > 0) timeString += minutes + "m ";
    timeString += seconds + "s";
    $('.timer').html(timeString);
}, 500);

// reset game listener
$(".restart").click(function() {
    initGame();
});





// initialize memory game
```

SUGGESTION

Great work with the commenting of the codes, its gives good information for a 3rd party reading the code. This submission could go further to beautify the comments and provide more detailed explanation for functions.

`/**

- @description Represents a book
- @constructor
- @param {string} title - The title of the book
- @param {string} author - The author of the book
- */

- function Book(title, author) {
- ...
- }`

More In-depth Knowledge

- You can follow the Udacity javascript code style style guide for better code style.
- http://usejsdoc.org/about-getting-started.html

```
function initGame() {
    let cards = shuffle(symbols);

    $(".stars").empty();
    for (var i = 0; i < 3; i++) {
        let star = $('<li><i class="fa fa-star"></i></li>');
        $(".stars").append(star);
    }
    $(".moves").text("0");
    // initial global vars
    open = [];
    moves = 0;
    match = 0;
    stars = 3;
    // activate timer
    startTime = new Date().getTime();

    // add event listeners to each card
    // handler function : clickCard()
    $(".deck").empty();
    for (var i = 0; i < cards.length; i++) {
        let card = $('<li class="card"><i class="fa fa-'+cards[i]+'"></i></li>')
        $(".deck").append(card);
        card.click(clickCard);
    }
}

// When all cards are successfully flipped,
// confirm dialog shows up giving gamer's performance
// in time usage and star level.
function endGame() {
```

SUGGESTION

Check the section titled *Comment* of this resource and provide better comments for the functions in this file.

```
    clearInterval(timer);
    let playAgain = confirm("Congratulations! You used " + ((new Date().getTime()-startTime)/1000).toFixed(2) + " seconds with " +
    if (playAgain) initGame();
}



/*
 * Display the cards on the page
 *   - shuffle the list of cards using the provided "shuffle" method below
 *   - loop through each card and create its HTML
 *   - add each card"s HTML to the page
 */

// Shuffle function from http://stackoverflow.com/a/2450976
function shuffle(array) {
    var currentIndex = array.length, temporaryValue, randomIndex;

    while (currentIndex !== 0) {
        randomIndex = Math.floor(Math.random() * currentIndex);
        currentIndex -= 1;
        temporaryValue = array[currentIndex];
        array[currentIndex] = array[randomIndex];
        array[randomIndex] = temporaryValue;
    }

    return array;
}


/*
 * set up the event listener for a card. If a card is clicked:
 *  - display the card's symbol (put this functionality in another function that you call from this one)
 *  - add the card to a *list* of "open" cards (put this functionality in another function that you call from this one)
 *  - if the list already has another card, check to see if the two cards match
 *    + if the cards do match, lock the cards in the open position (put this functionality in another function that you call from
 *    + if the cards do not match, remove the cards from the list and hide the card"s symbol (put this functionality in another fu
 *    + increment the move counter and display it on the page (put this functionality in another function that you call from this
 *    + if all cards have matched, display a message with the final score (put this functionality in another function that you cal
 */

```

```
// helper function
// compare whether two flipped cards match
function getSymbol(card) {
    return card.children().attr('class').substring(6);
}

// handler
function clickCard() {
```

SUGGESTION

Check the section titled *Comment* of this resource and provide better comments for the functions in this file.

```
    let currentCard = $(this);
    currentCard.addClass('open show');

    // first card in the "two" group
    if (open.length == 0) {
        open.push(currentCard);
    }
    // second card in the "two" group
    else {
        // timeout is 500 ms
        setTimeout(function() {
            moves++;
            currentCard.removeClass('open show');
            open[0].removeClass('open show');
            if (getSymbol(currentCard) == getSymbol(open[0])) {
                match++;
                currentCard.addClass('match');
                open[0].addClass('match');
            }
            open.pop();
            $(".moves").text(moves);

            if (moves == 15) {
                $(".stars").children().last().remove();
                stars--;
            }
            if (moves == 30) {
                $(".stars").children().last().remove();
                stars--;
            }
            if (moves == 45) {
                $(".stars").children().last().remove();
                stars--;
            }
            if (match == 8) {
                endGame();
            }
        }, 500);

    }
}

// main function
initGame()
```

▸ README.md 1

▸ index.html

▸ css/app.css

▸ CONTRIBUTING.md

Learn the best practices for revising and resubmitting your project.

RETURN TO PATH

Student FAQ