PromineoTech Week 6 Notes

```
# Author: William Z Chadwick
# Date Created: 09/13/2022
# Date Modified: 09/13/2022
# Description: Notes and Pseudo-Code for Promineo Tech Week 6 JavaScript War Card Game Project


---


---***---


---


Prompt:


NOTE: All of these details are included in the Coding Assignment Document


Preview:
This final JavaScript project is an automated version of the classic card game WAR.
	-As a part of this assignment, you will also be asked to consider:
		--Creating classes such as Card, Deck & Player
		--Keeping in mind what fields and methods each class might have.


	-The completed project will do the following:
		--Deal 26 Card to 2 Players from a Deck
		--Iterate through the turns where each Player plays a Card
			--Award a point to the Player with the higher Card
			--Ties result in zero points for both Players
		--After all cards have been played, display teh score and declare the winner


	-Finally, you will be asked to write a Unit Test using Mocha and Chai for at least one of your
functions


---


flow:


function playWar() {
	Class Deck {
		Class Card {
			. . .;
		}
	}
	Class Player {
		Class Hand {
			inherits Class card?;
			cards stored in Hand array;
		}
		has: spoils; spoils keeps count?
	}
	steps: {
		-fill out deck with (cards x13) x (suites x4), [points?]
		-instantiate 2 player objects and deck
		-shuffle deck
		-loop through shuffled deck, dealing every card to each player, alternating back and
forth
		-when both players have full 26 cards, begin play
			--play involves each player playing top card
				---1, add played card to player score/spoils.
				---2, use .pop() to take that off the top of the array?
				---exception: if card1Value = card2Value, neither score is added to player
score/spoils)
				---loop continues until hand1 + hand2 = 0;
			--when all cards are dealt, played, and scores added up; scores must be compared;
winningScore > losingScore, || if score1 === score2, game is tied.
	}
}
```

playWar();

note cards:

1
2
3
4
5
6
7 Do I need a separate function

Next document
Author: William Z Chadwick
Date Created: 09/07/2022
Date Modified: 09/07/2022
Description: A place for notes for my Promineo Tech, Week 6 video notes

---

Dev Tools; Chrome DevTools for debugging; first video

1, device toggle, elements, sources, applications tabs

right click;
choose inspect

1, device toggle
(can show mobile and desktop view and contents)

2, console
(console.log())

3, elements
(can help with troubleshooting)

4, Sources
(can look here to make sure it is picking up on your other files, instead of failing to find them, improper or missing links, etc.; can also take a look at code and what is actually happening

5, Network
6, Performance

7, Memory

8, Application
(caching anything, can see their values at this *key-value table* in the application tab.)

9, Security

10, Audits

---

end of video 1

---

beginning of video 2, Debugging

tools for debugging.

javascript
console.log
true true false false false

```
console.log(hasStringAtEnd('hello', 'llo')); // true
console.log(hasStringAtEnd('llo', 'hello')); // true
console.log(hasStringAtEnd('llod', 'hello')); // false
console.log(hasStringAtEnd('ll', 'hello')); // false
console.log(hasStrongAtEnd('llo', 'hellod')); // false

function hasStringAtEnd(a, b) {
        let shortest = '';
        let longest = '';
        if (a.length < b.length) {
                shortest = a;
                longest = b;
        } else {
                shortest = b;
                longest = a;
        }

        const indexStart = longest.length - shortest.length;
        const endOfLongest = longest.substring(indexStart + 1);
        return shortest === endOfLongest;
}
```

Next Document
// javascript test doc
//

```
console.log(hasStringAtEnd('hello', 'llo')); // true
console.log(hasStringAtEnd('llo', 'hello')); // true
console.log(hasStringAtEnd('llod', 'hello')); // false
console.log(hasStringAtEnd('ll', 'hello')); // false
console.log(hasStrongAtEnd('llo', 'hellod')); // false

function hasStringAtEnd(a, b) {
        let shortest = '';
        let longest = '';
        if (a.length < b.length) {
                shortest = a;
                longest = b;
        } else {
                shortest = b;
                longest = a;
        }

        const indexStart = longest.length - shortest.length;
        const endOfLongest = longest.substring(indexStart + 1);
        return shortest === endOfLongest;
}
```

PlayWarDurchLauf
// Author: William Z Chadwick
// Date Created: 09/14/2022
// Date Modified: 09/14/2022
// Description: A place for my notes following along with K Macias' video walk-through of Promineo Tech Week 7 playWar card program.
//

```
class Character {
        constructor(name, spoils, ) {}

}

class Card {
        constructor() {
```

```
                this.value = value;
                this.suit = suit;
                this.rank =
                // need rank and value; value = A, J, Q, K; rank = actual numerical value, 1-13;
        }
}

class Deck {}

class Character {}

let player1 = new Character('Kyle');
console.log(player1);
let player2 = new Character('Joe');
console.log(player2);

let values = ["2", "3", "4", "5", "6", "7", "8", "9", "10", "J", "Q", "K", "A"];
let suits ["Hearts", "Diamonds", "Spades", "Clovers"];
let ranks = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14];

createDeck() {
        for (let valueIndex = 0; valueIndex < values.length; valueIndex++) {
                for (let suitIndex = )

        }
}
dealDeck() {
        player1.hand = this.deck.slice(0, 26);
        console.log("printing player1 hand:", player1.hand);
        player2.hand = this.deck.slice(26, 52);
        console.log("printing player2 hand:", player2.hand);
        // slice method ignores the first number, so 26 and 0 are ignored
}

newDeck.createDeck();
newDeck.shuffleDeck();
newDeck.dealDeck();

// class for game

class Game {
        constructor() {

                // don't always have to utilize a constructor;
        }
        // we need a way to make 26 rounds happen; 1 card per round; which played card rank is bigger
than the other; assign point values, keep in mind;
        // so we need to 1. iterate 26 rounds
        // (use for-loop)
        // 2. a way to compare values of individual cards each roung
        // 3. assign point to score for player who won the round
        compareCards() {
                console.log(player1.score); // test
                console.log("testing compareCards method"); // test
                for (let round = 0; round < 26; round++) {
                        console.log(round);
                        console.log(player1.hand[round]);
                        // if else statements to check card values and determine round winner
                        console.log("player 1 played this card: ", player1.hand[round]);
                        console.log(player2.hand[round]);


                        if (player1.hand[round].rank > player2.hand[round].rank){console.log(player 1
wins the round!)
                                console.log("
                                        Round ${round}
                                        ${player1.hand[round].value} of ${player1.hand[round].suit}
                                        ${player2.hand[round].value} of ${player2.hand[round].suit}
");
                                console.log();
```

```
                                console.log();

                        } else if (player2.hand[round].rank > player1.hand[round].rank){
                                console.log("player 1 wins the round!");
                                console.log("player 1 played this card:");
                                console.log("player 2 played this card:");
                                console.log("
                                        Round ${round}
                                        ${player1.hand[round].value} of ${player1.hand[round].suit}
                                        ${player2.hand[round].value} of ${player2.hand[round].suit}
");


                        } else if (player1.hand[round].rank === player2.hand[round].rank) {
                                console.log();
                                console.log("
                                        Round ${round}
                                        ${player1.hand[round].value} of ${player1.hand[round].suit}
                                        ${player2.hand[round].value} of ${player2.hand[round].suit}
");
                                console.log()
                        };

                }
        }

}

let newGame = new Game();
newGame.compareCards();

// need final conditional comparing final spoils to each other;

// make it work, test it, clean it up;
```

```
PlayWar
// Author: William Z Chadwick
// Date Created: 09/14/2022
// Date Modified: 09/14/2022
// Description: My JavaScript version of the classic card game, "War".
//

//

//

// blog notes: good abstraction can be pragmatic; you are overwhelmed; you say, breathe. you say,
what is the next right thing. You put everything else out of your mind. You do the next right thing.
This is a form of good abstraction, and perhaps like methodical reductionism regarding steps to do in
the immediate future. Whereas, in Kierkegaard's words, "Freedom's possibility announces itself in
anxiety" (Concept of Anxiety, ch II.2.b, page 91,).

// what do I need to do next? I need to figure out everything that needs to go into the players
class.
class Player{
        constructor(name, hand, spoils) {
                this.name = name;
                this.hand = [];
                this.spoils = [];
        }
        //any additional methods?
        }
class Deck{ //cards
        constructor()
        this.deck = [];
}

let suites = ["Hearts", "Spades", "Diamonds", "Clubs"];
console.log(suites);
console.log(suites);
let value = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14];
```

```
console.log(value);
let rank = ["2", "3", "4", "5", "6", "7", "8", "9", "10", "J", "Q", "K", "A"];
console.log(rank);
// suites, value and rank have been console.log tested;
// now I should move over to working on Week 7 items;

class Card() {
        // so there should be a loop in the constructor, or a loop that calls the constructor 26
times? How did Macias fill this out in her walkthrough?
        let this.rank = rank;
        let this.suit = suit;
        let this.value = value;}
class Hand{
        // should hand be a sub-class of player?
}

// there needs to be a loop which fills out the deck with one of each of the suites, values, and
ranks;

let player1 = new Player("Joeski", [0], [0]);
console.log(player1);
let player2 = new Player("Broski", [0], [0]);
console.log(player2);

function playWar() {



        if {
        } else if {
        } else if {
};

playWar();
```

```
ShortestLongExample
console.log(hasStringAtEnd('hello', 'llo'));
console.log(hasStringAtEnd('llo', 'hello'));
console.log(hasStringAtEnd('llod', 'hello'));
console.log(hasStringAtEnd('ll', 'hello'));
console.log(hasStrongAtEnd('llo', 'hellod'));

function hasStringAtEnd(a, b) {
        let shortest = '';
        let longest = '';
        if (a.length < b.length) {
                shortest = a;
                longest = b;
        } else {
                shortest = b;
                longest = a;
        }

        const indexStart = longest.length - shortest.length;
        const endOfLongest = longest.substring(indexStart + 1);
        return shortest === endOfLongest;
}
```

```
# Author: William Z Chadwick
# Date Created: 09/07/2022
# Date Modified: 09/07/2022
# Description: A place for my notes during class in Week 6 of Promineo Tech, discusses Week 5 Project
some.

---

"Rubber-Ducking"

Rubber duck debugging.

talking about it to become more comfortable.
```

very cool stuff;

a web application version of the Week 6 version?

Sara Kuenzi is sharing her coding projects; haha, it's in really good shape for both weeks! cest la vie

I have got to find out how to devote more time to this.

I should check out Kuenzi's github for Week 5.

good to know that Kuenzi also followed the video; when asked if she plotted it out beforehand.

I should plot more out about that project. I really need to figure it out.

---

dan Wilson, made second menu also a while loop. // good idea

Pat W, asking how to use filter instead of splice (Sara used splice, he said).

Stepanski showing us an example from week 4; this is his example but with id key-value pairs added.

```
let parks = [
        { id: 1, name: "sd", rating: 4.2 }, // 4 items
        { id: 2, name: "hs", rating: 5 },
        { id: 3, name: "lw", rating: 4.5 },
        { id: 4, name: "js", rating: 4.1 },
];

// let result = parks.filter((park) => park.rating >= 4.5); // original example

let result = partks.filter((park) => park.name != "sd"); // tonights new code

console.log(result);
```

---

showing us the miro.com/ application for mapping coding thought. pretty cool. I made an account.

A pattern for engineering.

menu --> methods

^

TaskList --> methods, and perhaps properties

^

task --> only properties

---

pattern for engineering:

"The Single Responsibility Principle" (principle? pattern?)

related to "Separation of Concerns."

How does Participation change the need for this in real life, when not doing code?

Or resonances?

poetry often means multiple things at the same time - not a separation of concerns, which is why it is so rich, resonant, participatory in richer realities.

So living in a rich reality depends on good abstraction / participation, bringing disparate items to

participate in each other and in the ultimate Tao.

[asked at end of class; separation of concerns and single responsibility principle; he said that yeah, they are pretty much tied together. If one item has more than one responsibility, then it will be clearer and probably work better if the concerns are separated into single responsibilities per item.]

Had a hard time paying attention about half-way through class today (for maybe 20 min. or so); stopped taking notes, which is surely partly why. Hm. But that made me feel dissatisfied. I wish I had stuck with it. I feel like I'm being left behind - an awful feeling.

Part of this feeling is surely because I am actually being challenged this week and last week. And so the coursework isn't a breeze this week and last week. :/ I need to give it more time.

```
_populate() {

        const suits = ['', '', '', ''];
        const ranks = ['A', '2', '3', '4', '5', '6', '7', '8', '9', '10','J', 'Q', 'K'];
        const values = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13];

// Look this up: replit.com/@stepanski/week6-Assignmentindex.js

// stepanski says shuffling code stole from replit website on how to shuffle a javascript array (link in slack)
}
```

break the program into smaller parts and it becomes easier and easier.

boilerplate your code to write each method and what goes inside of it, separate everything into its separate parts, simple as possible.

// ** pseudo-code **

makes it easier to do.

"If you don't finish it, that's okay. Get through as much of it a you can. Get it to work. Understand how it works..."

end of class.

Test.js

```
// doc for testing purposes:
//


let suites = ["Hearts", "Spades", "Diamonds", "Clubs"];
console.log(suites);
let value = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14];
console.log(value);
let rank = ["2", "3", "4", "5", "6", "7", "8", "9", "10", "J", "Q", "K", "A"];
console.log(rank);
```

```
# Author: William Z Chadwick
# Date Created: 09/07/2022
# Date Modified: 09/07/2022
# Description: A place for my to-do list for Week 6 of Promineo Tech's Front End Bootcamp

1. extra resources:
-react + redux course sololearn
-same but video course
-more odin project
-post German notes to Josh
/-short review of "Writing a Book: Venturing into other fields in tech as a dev" codementor events video, blog post
-Valley of Code, React book, blog post

2. Week 6
-videos
-class attendance
```

	-quiz -project -** Get Ahead ** 3. Week 7 -research: semantic html tags x3 4. Week 8 CSS? 5. Week 5 completion -finish/perfect week 5 menu project -finish flash cards for bash and cpp -write about abstraction sources