WarPlus Design Specification

Table of Contents

[Summary 4](#_Toc66192867)

[Rules for the card game War 4](#_Toc66192868)

[THE DEAL 4](#_Toc66192869)

[THE PLAY 4](#_Toc66192870)

[HOW TO KEEP SCORE 4](#_Toc66192871)

[Pseudocode Game Flow 5](#_Toc66192872)

[Functions 7](#_Toc66192873)

[initializeGame() 7](#_Toc66192874)

[renderGame() 7](#_Toc66192875)

[shuffleDeck() 7](#_Toc66192876)

[splitDeck() 7](#_Toc66192877)

[showPayer1AmmoPile() 7](#_Toc66192878)

[hidePlayer1AmmoPile() 7](#_Toc66192879)

[showPlayer2AmmoPile() 7](#_Toc66192880)

[hidePlayer2AmmoPile() 7](#_Toc66192881)

[showPlayer1WinPile() 7](#_Toc66192882)

[hidePlayer1WinPile() 7](#_Toc66192883)

[showPlayer2WinPile() 7](#_Toc66192884)

[hidePlayer2WinPile() 7](#_Toc66192885)

[playPlayer1CardFaceUp() 8](#_Toc66192886)

[playPlayer2CardFaceUp() 8](#_Toc66192887)

[playPlayer1CardFaceUpFlipAnimation() 8](#_Toc66192888)

[playPlayer2CardFaceUpFlipAnimation() 8](#_Toc66192889)

[compareCards() 8](#_Toc66192890)

[moveTieCardsAside() 8](#_Toc66192891)

[playPlayer1CardFaceDown() 8](#_Toc66192892)

[playPlayer2CardFaceDown() 8](#_Toc66192893)

[moveCardsToPlayer1WinPile() 8](#_Toc66192894)

[moveCardsToPlayer2WinPile() 8](#_Toc66192895)

[movePlayer1WinPileToAmmoPile() 8](#_Toc66192896)

[movePlayer2WinPileToAmmoPile() 8](#_Toc66192897)

[Sounds 9](#_Toc66192898)

[shuffleSound 9](#_Toc66192899)

[playApplause 9](#_Toc66192900)

[playStartupSound 9](#_Toc66192901)

[playStartupSound2 9](#_Toc66192902)

[playClearTheBoardSound 9](#_Toc66192903)

[playCowbellSound 9](#_Toc66192904)

[playEnthusiasticApplauseSound 9](#_Toc66192905)

[playQuitSound 10](#_Toc66192906)

[playStartGameSound 10](#_Toc66192907)

[playSwish1Sound 10](#_Toc66192908)

[playSwish2Sound 10](#_Toc66192909)

[playSoSadSound 10](#_Toc66192910)

[Images 11](#_Toc66192911)

## Summary

This is the card game called War, implemented with a computer player, and with bells and whistles added.

## Rules for the card game War

**Players:** [2](https://bicyclecards.com/players/2/)

The goal is to be the first player to win all 52 cards

### THE DEAL

The deck is divided evenly, with each player receiving 26 cards, dealt one at a time, face down. Anyone may deal first. Each player places their stack of cards face down, in front of them.

### THE PLAY

Each player turns up a card at the same time and the player with the higher card takes both cards and puts them, face down, on the bottom of his stack.

If the cards are the same rank, it is War. Each player turns up one card face down and one card face up. The player with the higher cards takes both piles (six cards). If the turned-up cards are again the same rank, each player places another card face down and turns another card face up. The player with the higher card takes all 10 cards, and so on.

### HOW TO KEEP SCORE

The game ends when one player has won all the cards.

## Pseudocode Game Flow

<<This is now obsolete; will update on March 9th.>>

states:

deck Class

- cardsNum

- deckCategory (cardsUnplayed, cardsWon)

player Class

- player1CardsRemaining, player1WinPile

- player2CardsRemaining, player2WinPile

winner

tieQueue

tieWinner

1. playIntroGameMusic();

2. renderGameIntroScene();

3. turnOffMusic();

4. howToPlay() button is "available".

5. Initialize() game.

a) Randomize deck,

b) Split into two decks,

c) Assign each half to a player.

6. shuffleCards().

a) Actual card deck is randomized during initialize; this is just a visualization animation for entertaiment value.

7. renderGame()

a) first two cards are moved to the battlefield.

b) call cardsCompare().

c) if they are equivalent {

d) cardsGone()? Quit();

d.1) Move tie cards to the left. tieQueue.push()

d.2) Next two cards are moved to the battlefield.

d.3) If too many ties to move new battleCards into normal location, push new

card load to the right.

d.4) If a player runs out of cards during a tie, they lose.

d.5) Call cardsCompare().

e) else

e.1) Hightlight winning card with blinking border, moveCardsToWinnerAnimation();

e.1.a tieQueue.pop() until empty.

e.2) cardsGone()? declareWinner(), quit().

else

e.3) return renderGame();

8. quit()

a) renderWinnerAnimation();

b) playAgain()?;

## Functions

initializeGame() {

console.log("\nGame Initialized.");

};

renderGame(); {

console.log("\nGame Rendered. ");

};

shuffleDeck() {

console.log("\nCards shuffled.");

};

splitDeck() {

console.log("\nCard deck has been split into two randomized stacks of cards, called Ammo Piles.")

};

showPayer1AmmoPile() {

console.log("\nPlayer 1's Ammo Pile Visible.");

};

hidePlayer1AmmoPile() {

console.log("\nPlayer 1's Ammo Pile Hidden. ");

};

showPlayer2AmmoPile() {

console.log("\nPlayer 2's Ammp Pile Visible. ");

};

hidePlayer2AmmoPile() {

console.log("\nPlayer 2's Ammo Pile Hidden. ");

};

showPlayer1WinPile() {

console.log("\nPlayer 1's Win Pile Visible.");

};

hidePlayer1WinPile() {

console.log("\nPlayer 1's Win Pile Hidden. ");

};

showPlayer2WinPile() {

console.log("\nPlayer 2's Win Pile Visible. ");

};

hidePlayer2WinPile() {

console.log("\nPlayer 2's Win Pile Hidden. ");

};

playPlayer1CardFaceUp() {

console.log("\nPlayer 1's next card is face up. ");

};

playPlayer2CardFaceUp() {

console.log("Player 2's next card is face up");

};

playPlayer1CardFaceUpFlipAnimation() {

console.log("\nPlayer 1's next card is face up with STYLE!");

};

playPlayer2CardFaceUpFlipAnimation() {

console.log("\nPlayer 2's next card is face up with STYLE!");

};

compareCards() {

console.log("\nCards have been compared. Either Player 1 or Player 2 won, or it was a tie. ");

};

moveTieCardsAside() {

console.log("\nTie cards have been slid left to make room for more cards. ");

};

playPlayer1CardFaceDown() {

console.log("\nPlayer 1 played a facedown card due to a tie.");

};

playPlayer2CardFaceDown() {

console.log("\nPlayer 2 played a facedown card due to a tie.");

};

moveCardsToPlayer1WinPile() {

console.log("\nPlayer 1 won! Moved cards to his Win pile.");

};

moveCardsToPlayer2WinPile() {

console.log("\nPlayer 2 won! Moved cards to his Win pile. ");

};

movePlayer1WinPileToAmmoPile() {

console.log("\nPlayer 1 ran out of cards! Move his Win pile over to his Ammo pile.");

};

movePlayer2WinPileToAmmoPile() {

console.log("\nPlayer 2 ran out of cards! Move his Win pile over to his Ammo pile. ");

};

## Sounds

/\* Sound functions \*/

### shuffleSound

function playShuffleSound() {

var audio1 = new Audio('sounds/shuffling.mp3');

audio1.loop = false;

audio1.play();

};

### playApplause

function playApplause() {

var audio2 = new Audio('sounds/cheer.mp3');

audio2.loop = false;

audio2.play();

};

### playStartupSound

function playStartupSound() {

var audioStartup = new Audio('sounds/launchswell0.mp3');

audioStartup.loop = false;

audioStartup.play();

};

### playStartupSound2

function playStartupSound2() {

var audioStartup2 = new Audio('sounds/launchswell1.mp3');

audioStartup2.loop = false;

audioStartup2.play();

};

### playClearTheBoardSound

function playClearTheBoardSound() {

var audioClearTheBoard = new Audio('sounds/cleartheboard.mp3');

audioClearTheBoard.loop = false;

audioClearTheBoard.play();

};

### playCowbellSound

function playCowbellSound() {

var audioCowbellSound = new Audio('sounds/cowbell.mp3');

audioCowbellSound.loop = false;

audioCowbellSound.play();

};

### playEnthusiasticApplauseSound

function playEnthusiasticApplauseSound() {

var audioEnthusiasticApplauseSound = new Audio('sounds/enthusiastic-cheer.mp3');

audioEnthusiasticApplauseSound.loop = false;

audioEnthusiasticApplauseSound.play();

};

### playQuitSound

function playQuitSound() {

var audioQuitSound = new Audio('sounds/quit.mp3');

audioQuitSound.loop = false;

audioQuitSound.play();

};

### playStartGameSound

function playStartGameSound() {

var audioStartGameSound = new Audio('sounds/startgame.mp3');

audioStartGameSound.loop = false;

audioStartGameSound.play();

};

### playSwish1Sound

function playSwish1Sound() {

var audioSwish1Sound = new Audio('sounds/swish1.mp3');

audioSwish1Sound.loop = false;

audioSwish1Sound.play();

};

### playSwish2Sound

function playSwish2Sound() {

var audioSwish2Sound = new Audio('sounds/swish2.mp3');

audioSwish2Sound.loop = false;

audioSwish2Sound.play();

};

### playSoSadSound

function playSoSadSound() {

var audioSoSadSound = new Audio('sounds/launchswell0.mp3');

audioSoSadSound.loop = false;

audioSoSadSound.play();

};

## Images

#### 6 player 1 images:

#### Default player 1 Image:

#### 4 player 2 Images:

#### Default Player 2 Image: