

## Activity Diagram - Mini Uno



```
let pcHand: HTMLDivElement = document.querySelector("#pcHand");  
let playerHand: HTMLDivElement = document.querySelector("#playerHand");  
const possibleColors: string[] = ["yellow", "blue", "green", "red"];  
const possibleNumbers: number[] = [1, 2, 3, 4, 5, 6, 7, 8, 9];  
let randomColorValue: string;  
let randomNumberValue: string;  
let currentCard: HTMLSpanElement = document.querySelector("#currentCard");  
let deck: HTMLSpanElement = document.querySelector("#deck");  
let pass: HTMLButtonElement = document.querySelector("#pass");  
let pcTurn: boolean = false;  
let passAllowed: boolean = false;  
let pcNoMatch: boolean = false;
```



prompt an User: "How many cards do you want to start the game with?"

↳ Speichern in let userPrompt: string



string von prompt umwandeln in number mit parseInt(string).  
↳ Speichern in eine Variable let cardsNumber



Funktion aufrufen, die die Startkarten für den User generiert:  
generatePlayerCards();



Funktion aufrufen, die die Startkarten für den PC generiert:  
generatePcCards();



Funktion aufrufen, die die erste (zufällige) Karte auf dem Ablagestapel (currentCard) generiert:  
createFirstCard();



## generatePlayerCards

```
function generatePlayerCards(): void {  
  let i: number = 0;  
  while (i < cardNumber) {  
    createRandomValues();  
    let card: HTMLSpanElement = document.createElement("span");  
    playerHand.appendChild(card);  
    card.textContent = randomNumberValue;  
    card.className = randomColorValue;  
    card.addEventListener("click", chooseCard);  
    i++;  
  }  
}
```

## generatePcCards

```
function generatePcCards(): void {  
  let i: number = 0;  
  while (i < cardNumber) {  
    createRandomValues();  
    let card: HTMLSpanElement = document.createElement("span");  
    pcHand.appendChild(card);  
    card.textContent = randomNumberValue;  
    card.className = randomColorValue;  
    card.style.color = "lightslategrey";  
    card.style.backgroundColor = "lightslategrey";  
    i++;  
  }  
}
```

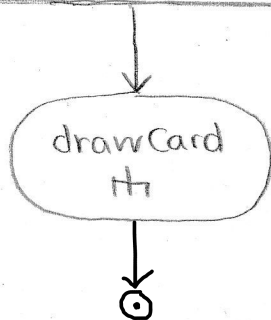
## createFirstCard

```
function createFirstCard(): void {  
  createRandomValues();  
  currentCard.textContent = randomNumberValue;  
  currentCard.className = randomColorValue;  
}
```

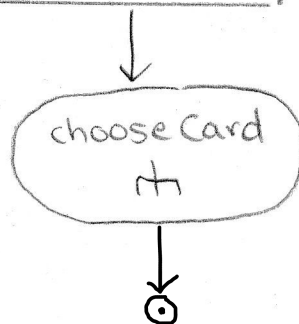
## handlePcTurn

```
function handlePcTurn(): void {  
  if (pcTurn) {  
    pcMove();  
  }  
}
```

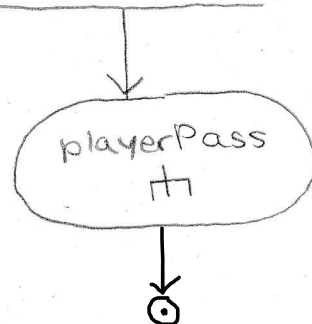
click on  
deck



click on  
card



click on  
pass



drawCard

-event: MouseEvent

```
let card: HTMLSpanElement = document.createElement("span");  
playerHand.appendChild(card);  
createRandomValues(); H  
card.textContent = "random Number Value";  
card.className = "random Color Value";  
passAllowed = true
```

playerPass

-event: MouseEvent

```
if (passAllowed == true) {  
  pcTurn = true;  
  handlePcTurn H  
}
```

chooseCard

-event: MouseEvent

```
let chosenCard: HTMLSpanElement = -event.target;  
if (chosenCard.textContent == currentCard.textContent ||  
    chosenCard.className == currentCard.className) {  
  currentCard.textContent = chosenCard.textContent;  
  currentCard.className = chosenCard.className;  
  playerHand.removeChild(chosenCard);  
  evaluateWinner(); H  
  pcTurn = true;  
  handlePcTurn(); H  
}
```

## CreateRandomValues

```
function createRandomValues(): void {  
    randomColorValue = possibleColors[Math.floor(Math.random()  
        * possibleColors.length)];  
    let x: number = possibleNumbers[Math.floor(Math.random()  
        * possibleNumbers.length)];  
    randomNumberValue = x.toString();  
}
```

## pcMove

```
function pcMove(): void {  
    let allCardsPc: HTMLCollection = pcHand.children;  
    pcNoMatch = true;  
    for (let i: number = 0; i < allCardsPc.length; i++) {  
        if (allCardsPc[i].textContent == currentCard.textContent ||  
            allCardsPc[i].className == currentCard.className) {  
            currentCard.textContent = allCardsPc[i].textContent;  
            currentCard.className = allCardsPc[i].className;  
            pcHand.removeChild(allCardsPc[i]);  
            passAllowed = false;  
            pcTurn = false;  
            pcNoMatch = false;  
            break;  
        }  
    }  
    if (pcNoMatch) {  
        pcDraw();  
    }  
}
```

evaluateWinner();

## pcDraw

```
function pcDraw(): void {  
    createRandomValues();  
    let card: HTMLSpanElement = document.createElement("span");  
    pcHand.appendChild(card);  
    card.textContent = randomNumberValue;  
    card.className = randomColorValue;  
    card.style.color = "lightslategrey";  
    card.style.backgroundColor = "lightslategrey";  
    if (card.textContent == currentCard.textContent ||  
        card.className == currentCard.className) {  
        currentCard.textContent = card.textContent;  
        currentCard.className = card.className;  
        pcHand.removeChild(card);  
    }  
    pcTurn = false;  
    passAllowed = false;  
}
```

## evaluateWinner

```
function evaluateWinner(): void {  
  let allCardsPc: HTMLCollection = pcHand.children;  
  let allCardsPlayer: HTMLCollection = playerHand.children;  
  
  if (allCardsPc.length == 0) {  
    alert("Du hast verloren :(");  
    location.reload();  
  }  
  
  if (allCardsPlayer.length == 0) {  
    alert("Du hast gewonnen");  
    location.reload();  
  }  
}
```