2025 Presentation

JS를 활용한 게임 구현

고희권

2025 presentation

게임설명

자르기 버튼을 눌러 샌드위치를 5번 자르면 성공입니다.

1. 샌드위치는 좌우로 움직여요.

2. 최대한 샌드위치가 중앙에 위치할 때 자르기 버튼을 누르세요.

3. 샌드위치가 잘리면 큰 부분은 버려지고 작은 부분이 남습니다.

4. 자르기를 성공할 때 마다 샌드위치의 속도가 소폭 증가합니다.

5. 5번 잘랐을 때 샌드위치가 남는 다면 스테이지 클리어

6. 다음 단계에 도전이 가능하며, 다음 단계에서는
   기본속도가 소폭 증가합니다.

게임구상

01

샌드위치 객체는 좌우로
움직여야 하며, 화면을
벗어날 수 없다.

02

칼은 중앙에 위치하고,
수직방향으로만 움직이며,
화면을 벗어날 수 없다.

03

샌드위치가 잘렸을 때 큰 부분은
버려지고, 작은 부분은 남긴다.

04

게임 시작화면: 닉네임, 시작, 도움말
게임화면: 샌드위치, 칼, 자르기
게임 결과화면: 다음단계, 다시하기
처음으로

05

반응형 CSS 작성 필요

주요기능

01 〈img〉는 〈div〉내부에 위치하며 같이 이동합니다.

샌드위치가 잘리게 되면 〈div〉값이 변경되며
〈div〉의 style.overflow = hidden으로 인해
잘린 부분의 이미지는 보이지 않고 비율이 유지됩니다.

샌드위치의 길이는 캔버스 크기에 따라 달라집니다.

this.vX의 값은 샌드위치의 이동속도이며
캔버스 크기와 단계 그리고 자르기 성공횟수에 따라 달라집니다.

자르기 성공횟수에 따른 속도는 단계가 넘어가면 초기화됩니다.

```
function Sandwich(){
    this.x = 0
    this.y = 200
    this.vX = parseInt(parseInt(mainDivStyle.width)/300)
        + parseInt(parseInt(mainDivStyle.width)/500)*(stage-1)
    this.height = parseInt(mainDivStyle.height)/3
    this.width = parseInt(mainDivStyle.width)/2 * 1.5
    this.div = document.createElement('div')
    this.div.style.position = "absolute"
    this.div.style.height = this.height + "px"
    this.div.style.width = this.width + "px"
    this.div.style.top = this.y + "px"
    this.div.style.zIndex = 3
    this.div.style.overflow = "hidden"
    this.bread = document.createElement('img')
    this.bread.style.position = "relative"
    this.bread.src = "./img/sandwich.png"
    this.bread.style.height = this.height + "px"
    this.bread.style.width = this.width + "px"
    this.bread.style.left =  "0px"
    this.bread.style.zIndex = 2
    this.div.appendChild(this.bread)
    document.querySelector("#canvas").appendChild(this.div)
}
```

주요기능

02 샌드위치의 move메소드

샌드위치의 move메소드와 샌드위치를 움직일 setInterval 부분

샌드위치 move메소드에서는
샌드위치가 캔버스를 벗어나는지 확인 및 방향 전환 후 vX만큼 이동합니다

빵이 움직이고 있는지 체크하기 위해 moving 변수를 사용
moving 값을 체크하여 빵의 이동을 시작합니다.
(추가설명 예정)

```
Sandwich.prototype.move = function(){
    let width = this.div.style.width
    let count=0
    if(parseInt(this.x) + parseInt(width)
        > parseInt(mainDivStyle.width) || this.x < 0){
        this.vX *= -1
    }
    this.x += this.vX
    this.div.style.left = this.x + "px"
}
sGame = setInterval(function() {
    if(moving == 0){
        moving = 1
        sandwich.vX > 0 ?
            sandwich.vX += stageSpeed
            : sandwich.vX -= stageSpeed //단계 별 속도 증가값
        sMove = setInterval(function() {
            sandwich.move()
            if(cuttingCount == 5){
                length = parseInt(sandwich.div.style.width)
                gameEnd(length, cuttingCount)
            }
        },1000/60)
    }else{
    }
},500)
```

주요기능

03 칼 객체와 cut 메소드

칼 객체는 항상 캔버스 중앙에 위치합니다.

칼의 넓이는 일정하나 높이는 캔버스 높이에 따라 달라집니다.

cut메소드는 샌드위치 move메소드와 작동방식이 동일합니다.

아래 위로 이동한 뒤 캔버스 최상단에 닿게 되면
리턴을 통해 cut Interval을 종료합니다.

```
function Sword(){
    this.x = parseInt(mainDivStyle.width)/2 - 10
    this.y = parseInt(mainDivStyle.height)/10
    this.vY = 25
    this.height = parseInt(mainDivStyle.height)/5
    this.width = 20
    this.sword = document.createElement('img')
    this.sword.style.position = "absolute"
    this.sword.src = "
    this.sword.style.h
    this.sword.style.v
    this.sword.style.l
    this.sword.style.t
    this.sword.style.z
    document.querySele
}
```

```
Sword.prototype.cut = function(height){
    if(this.y + height > 300 ){
        this.vY*= -1
    }else if(this.y < 50) {
        this.y = 60
        this.vY = 20
        this.sword.style.top = this.y + "px"
        cutting = 0
        return clearInterval(cut)
    }
    this.y += this.vY
    this.sword.style.top = this.y + "px"
}
```

주요기능

04 샌드위치 자를 때 남길 부분의 계산식

칼로 잘렸을 때 왼쪽의 길이는 L, 오른쪽의 길이는 R로 표현
샌드위치의 길이는 W, 캔버스 중앙은 C, 빵의 위치는 X로 표현

오른쪽이 남는 상황 R = (X + W) − C
왼쪽이 남는 상황 L = C − (X)

각 값으로 〈div〉의 width를 변경
이 때 오른쪽이 남는 상황에서는 〈img〉의 left와 x 변경 필요
x를 변경하지 않으면 자른 후의 위치가 왼쪽으로 이동하는 버그 발생
left를 변경하지 않으면 항상 왼쪽 샌드위치만 남는 버그 발생

해당 부분은 게임을 실행 후 코드와 같이 보여드리겠습니다.

```
document.querySelector("#cut_btn").addEventListener('click',function(){
    if(cutting == 0){
        cutting = 1
        cut = setInterval(function(){
            sword.cut(sword.height)
        },1000/60)
        let dLeft = parseInt(sandwich.div.style.left)
        let sLeft = parseInt(sandwich.bread.style.left)
        let width = parseInt(sandwich.div.style.width)
        let cavasCenter = canvasWidth/2
        let cC = canvasWidth/2
        let sC = dLeft + width/2
        if(sC >= cC){
            if((cC-dLeft) > 0){ sandwich.div.style.width = (cC-dLeft) + "px"}
            else{           sandwich.div.style.width = 0 + "px"          }
        }else{
            if((dLeft + width - cC) > 0){
                sandwich.div.style.width = (dLeft + width - cC) + "px"
                sandwich.div.style.left = cC + "px"
                sandwich.x = cC
                sandwich.bread.style.left = sLeft + ((cC - dLeft)*-1) + "px"
            }else{           sandwich.div.style.width = 0 + "px"          }
        }
```

주요기능

05 setInterval IF분기를 위한 변수 사용

1. GameMove Interval 실행 및 moving값 확인
   1-1. moving==0 이라면
      1-1-1. sandwich Move interval 실행
   1-2. moving==1 일 때는 동작 건너뛰기

2. 자르기 버튼 클릭시 이벤트 함수 실행
   2-1. cutting==0 이라면 cut Interval실행
      2-1-1. sMove 종료 후 cutting 메소드 실행
            cutting 메소드 종료시 moving=0
   2-2. cutting==1 이라면 동작 건너뛰기

```
//샌드위치 무빙 시작
sGame = setInterval(function() {
    if(moving == 0){
        moving = 1
        sandwich.vX > 0 ?
            sandwich.vX += stageSpeed
            : sandwich.vX -= stageSpeed //단계 별 속도 증가값
        sMove = setInterval(function() {···
    }else{
    }
},500)

//클릭시 커팅 시작
let sLeftPx = 0
document.querySelector("#cut_btn").addEventListener('click',function(){
    if(cutting == 0){···
    clearInterval(sMove)
    moving = 0
})
```

주요기능

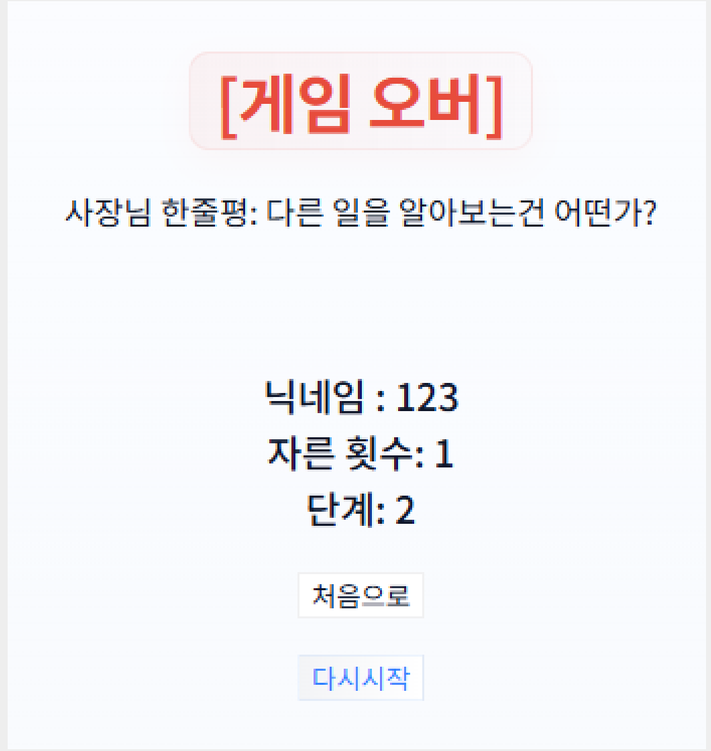
06 기타 화면

게임시작 화면과 게임 결과 화면입니다.

게임시작 화면에서는 닉네임 입력, 도움말, 게임시작 버튼으로 구성했습니다.

게임결과 화면은 성공 실패 여부에 따라 화면이 출력됩니다.

사장님 한줄평은 각 상황에 따라 3개 중 1개가 랜덤으로 출력됩니다.



locastarage: 브라우저에 값을 저장하는 객체
setItem: 변수에 값을 저장하기 위한 메소드, locaStorage.setItem("변수명",저장할 값)
getItem: 저장한 변수를 불러오기 위한 메소드,let 변수명 = locaStorage.getItem("변수명")

닉네임, 남긴 길이, 단계, 자른 횟수 등을 활용하기 위해 해당 객체를 사용했습니다.

결과평가

원하는 기능을 구현하고 게임 프로젝트를 완료했습니다.

프로젝트를 진행하며 좋았던 점은 원하는 기능을 구현하기 위해
어떤 태그를 사용해야하는지 생각하고 찾아보고 구현하며
공부를 할 수 있었던 점이 좋았습니다.

다만 아쉬운 점으로는 계획을 완료하지 않고 코드를 작성하여 유지보수가 힘든 점입니다.
중간에 기능을 추가하고, 반응형으로 변경하다 보니 변수와 CSS스타일 값들이
객체와 함수에 흩어져 있어 원하는 기능을 반영하는데 많은 시간이 소요됐고
예기치 못한 버그들이 발생했습니다.

다음 프로젝트를 진행하게 될 때에는 계획을 잘 세워
유지보수 및 기능추가가 용이한 코드를 작성해야되겠다고 생각했습니다.