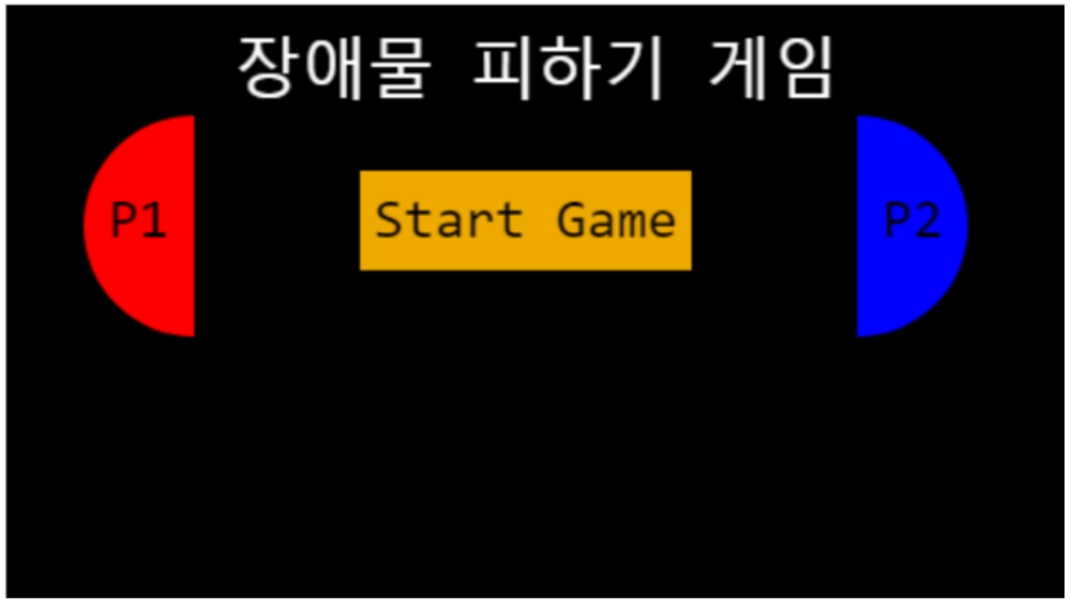
게임프로그래밍

HTML Game Upgrade 내용 발표

2019775009
소프트웨어학과
김수민

미리보기


SCORE: 1193


장애물 피하기 게임
P1
Start Game
P2


장애물 피하기 게임
Easy
Normal
Hard

추가된 기능들

1.다중 키 입력

2.플레이어 이미지 3가지 형태 변환

3.장애물 랜덤 색상 추가

4.인트로 추가

5.배경음,효과음 추가

6.멀티플레이 추가(1p,2p)

7.난이도 선택

8.재 시작 버튼 추가

9.배경이미지 추가

```
var myGamePiece;                  // 플레이어가 조작하는 객체(사각형)를 참조하는 변수
var myObstacles = [];             // 장애물을 저장하는 배열
var myScore;                      // 점수를 저장하는 변수

var mySound;                      // 효과음 추가
var clicked = false;              // 마우스 클릭 여부
var myBackground;                 // 배경추가
var myMusic;                      // 배경음 추가
var player2_Clicked    = false;   // 플레이어 2 체크 여부
var player1_Clicked    = false;   // 플레이어 2 체크 여부
var started_flag = false;         // StartGame 버튼 클릭 여부
var level_flag2 = false;          // 난이도 버튼 클릭 여부
var setSpeed = 1;                 // 난이도별 스피드
var setinterval = 150;            // 난이도별 장애물 생성기준 단축 기존 150
var sounds = ["Easy.mp3","normal.mp3","Extreme.mp3"]; // 난이도별 배경음
```

각 기능에 특이사항

1.인트로 화면에서 플레이어 선택없이 게임시작 못 합니다

2.배경음은 각 난이도에 따라 달라집니다

3.난이도별 장애물 생성 속도 증가합니다

4.난이도별 유저와 장애물의 속도 증가합니다.

5.재시작 버튼은 게임이 종료되었을 때만 보여집니다

6. 플레이어 3가지 형태 이미지

인트로 구현 코드(1)


```
intro = new component("25px","Consolas", "black",235,105,"intro");
intro2 = new component("25px","Consolas", "black",480,270,"intro2");
intro.update();
myName.text="장애물 피하기 게임";
myName.update();
```

MyGameArea 객체 내에서
각 flag 변수는 동이한 좌표값 중복을 방지했습니다

순서 (1)

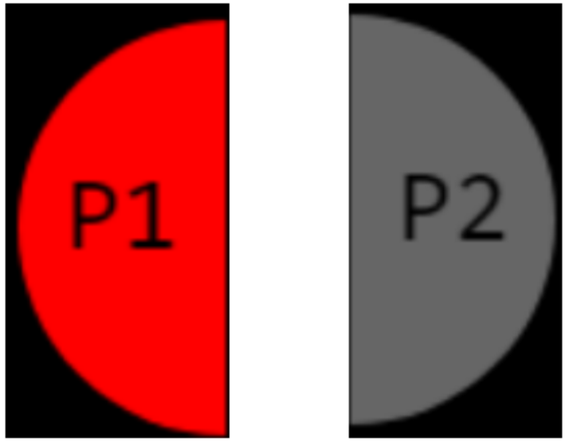
```
this.canvas.addEventListener("click", function(event) {
this.x = event.clientX ;
this.y = event.clientY ;

if(!started_flag){
    changeBtnColor(this.x,this.y,85,100,385,100);
    // StratGame을 눌렀을 때
    if(this.x >= 235-70 && this.x <= (235-70)+150
        && this.y > 105-25 && this.y <= (105-30)+55)
        {
            if(!player1_Clicked && !player2_Clicked ){
                alert("플레이어를 선택하세요");
            }else{
                started_flag = true;
                intro2.update();
            }

        }
```

순서 (2,3)

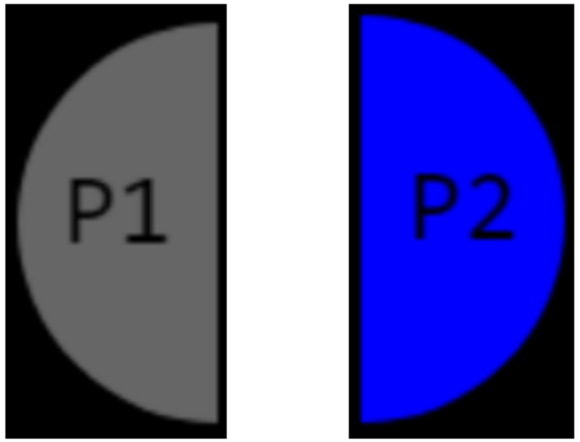
```
}else if(!level_flag2){
    // 쉬움 버튼 영역 확인
    if(this.x > ((480-300)/2) && this.x < ((480-300)/2 + 70) &&
        this.y > ((270-30)/2) && this.y < ((270-30)/2 + 70)) {
            myMusic = new sound(sounds[0]);
            level_flag2 = true;
            setinterval = 150;
            setSpeed = 1;
            myMusic.play();
            self.interval = setInterval(updateGameArea, 20);
    }
    // 노말 버튼 영역 확인
    else if(this.x > ((480-70)/2) && this.x < ((480-70)/2 + 70) &&
    this.y > ((270-30)/2) && this.y < ((270-30)/2 + 70)) {
            myMusic = new sound(sounds[1]);
            level_flag2 = true;
            setinterval = 60;
            setSpeed = 3;
            myMusic.play();
            self.interval = setInterval(updateGameArea, 20);
    }
    // 하드 버튼 영역 확인
    else if(this.x > ((480/2)+80) && this.x < ((480/2)+80 + 70) &&
    this.y > ((270-30)/2) && this.y < ((270-30)/2 + 70)) {
            myMusic = new sound(sounds[2]);
            level_flag2 = true;
            setinterval = 30;
            setSpeed = 5;
            myMusic.play();
            self.interval = setInterval(updateGameArea, 20);
    }

}
});
```

인트로 구현 코드(2)

P1 버튼 클릭 했을 때



P2 버튼 클릭 했을 때




```
// 각 버튼에 대해 클릭하면 클릭하지 않은 버튼의 색깔은 바뀝니다
function changeBtnColor(x,y,p1_x,p1_y,p2_x,p2_y){
    // P1 버튼
    var dx1 = x - p1_x;
    var dy1 = y - p1_y;
    var angle1 = Math.atan2(dy1, dx1);
    if (dx1 * dx1 + dy1 * dy1 <= 50 * 50 && (angle1 > Math.PI / 2 || angle1 < -Math.PI / 2)) {
        player1_Clicked = true;
        player2_Clicked = false;
        intro.color2 ="#666666";
        intro.color = "#FF0000";
        myGameArea.clear();
        intro.update();
        myName.update();
    }
    // P2 버튼
    var dx2 = x - p2_x;
    var dy2 = y - p2_y;
    var angle2 = Math.atan2(dy2, dx2);
    if (dx2 * dx2 + dy2 * dy2 <= 50 * 50 && (angle2 < Math.PI / 2 && angle2 > -Math.PI / 2)) {
        player1_Clicked = false;
        player2_Clicked = true;
        intro.color2 ="#0000FF";
        intro.color = "#666666";
        myGameArea.clear();
        intro.update();
        myName.update();
    }
```

인트로 구현 코드(3)

component 객체에서 인트로에 사용되는 요소들을 생성하였습니다.


```
else if (this.type == "intro2") {
        // 이전 인트로 제거
        myGameArea.clear();
        // 백그라운드 배경
        ctx.fillStyle = color;
        ctx.fillRect(0,0,myGameArea.canvas.width,myGameArea.canvas.height);
        myName.update();

        // 쉬움 배경,버튼
        ctx.fillStyle = "#00FF33";
        ctx.fillRect((480-300)/2,(270-30)/2,70,70);

        ctx.fillStyle = "#00FF33";
        ctx.textAlign = 'center';
        ctx.font = "30px", "Consolas";
        ctx.fillText("Easy",(480-230)/2, ( (270-30) /2 )-20 );


        // 노말 배경,버튼
        ctx.fillStyle = "#FFCC00";
        ctx.fillRect((480-70)/2,(270-30)/2,70,70);

        ctx.fillStyle = "#FFCC00";
        ctx.textAlign = 'center';
        ctx.font = "30px", "Consolas";
        ctx.fillText("Normal",(480)/2, ( (270-30) /2 )-20 );


        // 하드 배경,버튼
        ctx.fillStyle = "#CC0000";
        ctx.fillRect( (480/2)+80 ,(270-30)/2,70,70);

        ctx.fillStyle = "#CC0000";
        ctx.textAlign = 'center';
        ctx.font = "30px", "Consolas";
        ctx.fillText("Hard",(480/2)+115, ( (270-30) /2 )-20 );
    }
    else if (this.type == "intro") {
        // 백그라운드 배경
        ctx.fillStyle = color;
        ctx.fillRect(0,0,myGameArea.canvas.width,myGameArea.canvas.height);

        // GameStart 버튼 배경
        ctx.fillStyle = "#eeaa00";
        ctx.fillRect(this.x-75,this.y-30,150,45);

        // GameStart 버튼
        ctx.fillStyle = color;
        ctx.textAlign = 'center';
        ctx.font = this.width + " " + this.height;
        ctx.fillText("Start Game",this.x,this.y);

        // plyaer1 배경
        ctx.beginPath();
        ctx.fillStyle = this.color;
        ctx.arc(85, 100, 50, 0.5 * Math.PI, 1.5 * Math.PI);
        ctx.stroke();
        ctx.fill();

        // Player 1 버튼
        ctx.fillStyle = color;
        ctx.textAlign = 'center';
        ctx.font = this.width + " " + this.height;
        ctx.fillText("P1",this.x-175,this.y);

        // plyaer2 배경
        ctx.beginPath();
        ctx.fillStyle = this.color2;
        ctx.arc(385, 100, 50, 1.5 * Math.PI, 0.5 * Math.PI);
        ctx.stroke();
        ctx.fill();

        // Player 2 버튼
        ctx.fillStyle = color;
        ctx.textAlign = 'center';
        ctx.font = this.width + " " + this.height;
        ctx.fillText("P2",this.x+175,this.y);
    }
```

키보드 입력 구현 코드

MyGameArea 객체 내에서

```
window.addEventListener('keydown', function (e) {
    myGameArea.keys = (myGameArea.keys || []);
    myGameArea.keys[e.keyCode] = true;
})
window.addEventListener('keyup', function (e) {
    myGameArea.keys[e.keyCode] = false;
})
```

updataGameArea 함수 내에서

```
// 키보드 입력 *마우스 클릭으로 유닛을 움직이지 않는 조건
if(!clicked) {
    keymove(setSpeed);
 };
```

```
// 키보드 입력 ,  난이도에 따라 속도 변환
function keymove(x){
if(player2_Clicked){
    clearmove();
    clearmove2()
    if (myGameArea.keys && myGameArea.keys[37]) {myGamePiece.speedX = -x; myGamePiece.image.src = "fighter_2.png"; }
    if (myGameArea.keys && myGameArea.keys[39]) {myGamePiece.speedX = x; myGamePiece.image.src = "fighter_2.png";}
    if (myGameArea.keys && myGameArea.keys[38]) {myGamePiece.speedY = -x; myGamePiece.image.src = "fighter_2.png";}
    if (myGameArea.keys && myGameArea.keys[40]) {myGamePiece.speedY = x; myGamePiece.image.src = "fighter_2.png";}

    if (myGameArea.keys && myGameArea.keys[87]) {myGamePiece2.speedY = -x; myGamePiece2.image.src = "fighter_2_2.png";
    if (myGameArea.keys && myGameArea.keys[83]) {myGamePiece2.speedY = x; myGamePiece2.image.src = "fighter_2_2.png";}
    if (myGameArea.keys && myGameArea.keys[65]) {myGamePiece2.speedX = -x; myGamePiece2.image.src = "fighter_2_2.png";
    if (myGameArea.keys && myGameArea.keys[68]) {myGamePiece2.speedX = x; myGamePiece2.image.src = "fighter_2_2.png";}
}else{
    clearmove();
    if (myGameArea.keys && myGameArea.keys[37]) {myGamePiece.speedX = -x; myGamePiece.image.src = "fighter_2.png"; }
    if (myGameArea.keys && myGameArea.keys[39]) {myGamePiece.speedX = x; myGamePiece.image.src = "fighter_2.png";}
    if (myGameArea.keys && myGameArea.keys[38]) {myGamePiece.speedY = -x; myGamePiece.image.src = "fighter_2.png";}
    if (myGameArea.keys && myGameArea.keys[40]) {myGamePiece.speedY = x; myGamePiece.image.src = "fighter_2.png";}
}
}
```

장애물 랜덤 색상 코드


```
// 16진수 랜덤 색상 추가
function getRandomColor() {
    var letters = '0123456789ABCDEF';
    var color = '#';
    for (var i = 0; i < 6; i++) {
        color += letters[Math.floor(Math.random() * 16)];
    }
    return color;
}
```

```
var color = getRandomColor(); // 랜덤 색상을 생성합니다.
myObstacles.push(new component(10, height, color, x, 0));
myObstacles.push(new component(10, x - height - gap, color, x, height + gap));
// 서로 다른 크기와 좌표값의 장애물들 생성
```

효과음,배경음 코드


```
// sound 객체 추가
function sound(src) {
  this.sound = document.createElement("audio");
  this.sound.src = src;
  this.sound.setAttribute("preload", "auto");
  this.sound.setAttribute("controls", "none");
  this.sound.style.display = "none";
  document.body.appendChild(this.sound);
  this.play = function(){
    this.sound.play();
  }
  this.stop = function(){
    this.sound.pause();
  }
}
```

```
for (i = 0; i < myObstacles.length; i += 1) {  // 장애물의 수 만큼 반복
    if (myGamePiece.crashWith(myObstacles[i])) { // 장애물에 하나라도 충돌 시 멈춤
        // 충돌 시 이미지 변환,효과음 발생,배경음 멈춤
        myMusic.stop();
        mySound.play();
        myGamePiece.image.src = "boom.png";
        myGamePiece.update();
        setTimeout(function() {
            myGameArea.stop();
        }, 100);
        return;
    }if(player2_Clicked){   // 2P일 때
        if (myGamePiece2.crashWith(myObstacles[i])) {
        myMusic.stop();
        mySound.play();
        myGamePiece2.image.src = "boom.png";
        myGamePiece2.update();
        setTimeout(function() {
            myGameArea.stop();
        }, 100);
        return;
     }
    }
}
```

```
myMusic = new sound(sounds[0]);
level_flag2 = true;
setinterval = 150;
setSpeed = 1;
myMusic.play();
```

```
myMusic = new sound(sounds[1]);
level_flag2 = true;
setinterval = 60;
setSpeed = 3;
myMusic.play();
```

```
myMusic = new sound(sounds[2]);
level_flag2 = true;
setinterval = 30;
setSpeed = 5;
myMusic.play();
```

게임 재시작 버튼 코드


```
stop : function() {
    // 인터벌 중지
    clearInterval(this.interval);
    document.getElementById('restart-button').style.display = 'block';
},
```

```
<div id="game-container">
    <button id="restart-button" onclick="Restart()" style="display: none;">Restart</button>
</div>
```

```
// 재랜더링
function Restart(){
    location.reload();
}
```

출처

https://www.w3schools.com/
https://pine-j.medium.com/how-to-html-canvas- buttons-396267c6e5b5
https://www.chosic.com/download-audio/28027/
https://pixabay.com/sound-effects/search/bounce/
https://icons8.com/
http://www.n2n.pe.kr/lev-1/color.htm

감사합니다