2022 –2 게임프로그래밍

HTML GAME – BLOCK

2022 – 2 GAME PROGRAMMING
PROJECT_2 : HTML GAME
GAME'S NAME : BLOCK GAME

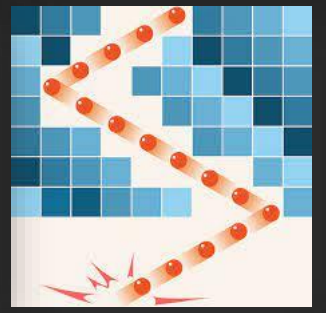
2020875042

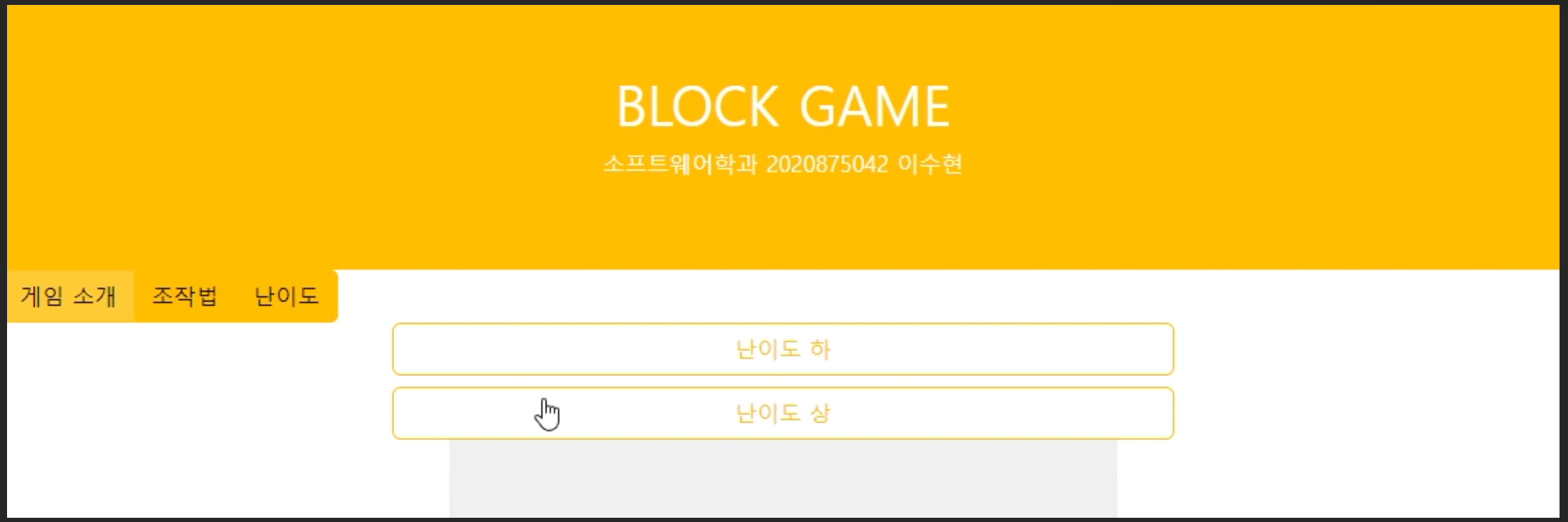
소프트웨어학과 이수현

01

BLOCK GAME 소개

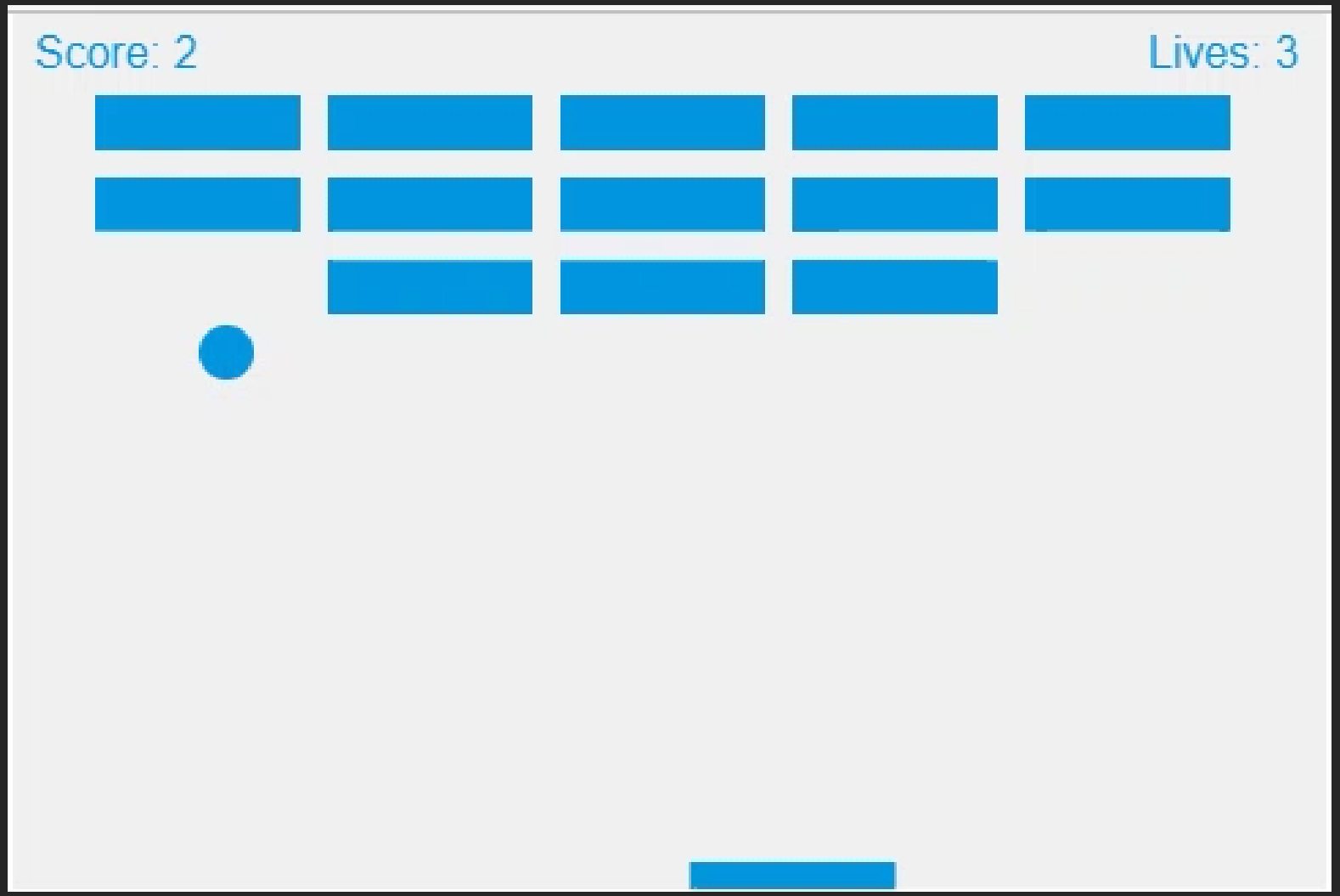
공이 움직이면서 벽돌을 깨는 게임입니다.
휴대폰 어플로도 많이 보셨을 듯 합니다.
아래는 간단한 시연 영상입니다.
시연 영상은 UPGRADE 한 최종 버전입니다.





02

BLOCK GAME ORIGINAL 시연



03

ORIGINAL CODE 설명

캔버스와 변수 선언을 위한 부분

```
13  <canvas id="myCanvas" width="480" height="320"></canvas>
14
15  <script>
16      // canvas 위에 그래픽을 렌더링 하기 위해서 js로 참조할 수 있도록 하는 것
17      let canvas = document.getElementById("myCanvas");
18      // 캔버스에 그리기 위해 실질적으로 사용되는 도구인 2D rendering context를 ctx 변수에 저장
19      let ctx = canvas.getContext("2d");
20      let ballRadius = 10;
21      let x = canvas.width/2;
22      let y = canvas.height-30;
23      let dx = 2;
24      let dy = -2;
25      // 공을 치기 위한 paddle 생성
26      let paddleHeight = 10;
27      let paddleWidth = 75;
28      let paddleX = (canvas.width-paddleWidth)/2;
29      let rightPressed = false;
30      let leftPressed = false;
31      let brickRowCount = 5;
32      let brickColumnCount = 3;
33      let brickWidth = 75;
        let brickHeight = 20;
        let brickPadding = 10;
        let brickOffsetTop = 30;
37      let brickOffsetLeft = 30;
38      let score = 0;
39      let lives = 3;
```

03

ORIGINAL CODE 설명

키보드와 마우스 제어를 위한 부분

키보드의 오른쪽, 왼쪽 방향키를 제어하기 위해
keyDownHandler, keyUpHandler 함수를 정의한다.

방향키를 눌렀을 때, keyDownHandler가 실행되고,
키에서 손을 뗐을 때, keyUpHandler가 실행된다.

다른 키에 대해서는 정의를 하지 않았으므로,
키를 누르더라도 다른 반응을 보이지 않는다.

키보드로만 플레이하는 것이 아니라, 마우스를 통해
플레이 할 수 있도록 mouseMoveHandler도 만든다.

```
50  // 키보드 중 어떤 키가 눌러지면 keydown 실행, 키를 떼면 keyup 실행
51  document.addEventListener("keydown", keyDownHandler, false);
52  document.addEventListener("keyup", keyUpHandler, false);
53  document.addEventListener("mousemove", mouseMoveHandler, false);
54
55  function keyDownHandler(e) {
56      if(e.key == "Right" || e.key == "ArrowRight") {
57          rightPressed = true;
58      }
59      else if(e.key == "Left" || e.key == "ArrowLeft") {
60          leftPressed = true;
61      }
62  }
63
64  function keyUpHandler(e) {
65      if(e.key == "Right" || e.key == "ArrowRight") {
66          rightPressed = false;
67      }
68      else if(e.key == "Left" || e.key == "ArrowLeft") {
69          leftPressed = false;
70      }
71  }
72
73  function mouseMoveHandler(e) {
74  let relativeX = e.clientX - canvas.offsetLeft;
75  if(relativeX > 0 && relativeX < canvas.width) {
76      paddleX = relativeX - paddleWidth/2;
77  }
78  }
```

03

ORIGINAL CODE 설명

벽돌과 공 충돌 제어를 위한 부분

벽돌과 공이 충돌되었을 때 벽돌이 없어져야 하기 때문에 해당 부분을 구현한 코드이다.

status 변수를 만들어, 충돌이 일어나기 전에는 1로, 충돌이 일어난 후에는 0으로 변경하게 만들어 벽돌을 없앨 수 있도록 되어있다.
상태가 0이 되면 점수가 증가하고, 점수가 총 벽돌 수와 같아졌을 때, 게임을 종료한다.

```
79    // 벽돌과 공 사이의 충돌을 감지한다.
80    // 조건 1. 공의 x 좌표는 벽돌의 x 좌표보다 커야 한다.
81    // 조건 2. 공의 x 좌표는 벽돌의 x 좌표 + 가로 길이보다 작아야 한다.
82    // 조건 3. 공의 y 좌표는 벽돌의 y 좌표보다 커야 한다.
83    // 조건 4. 공의 y 좌표는 벽돌의 y 좌표 + 높이보다 작아야 한다.
84
      // 충돌이 일어나기 전에는 상태 1로, 충돌 이후에는 0으로 변경해 벽돌이 없어지도록 한다.
      function collisionDetection() {
      for(let c=0; c<brickColumnCount; c++) {
          for(let r=0; r<brickRowCount; r++) {
          let b = bricks[c][r];
          if(b.status == 1) {
              if(x > b.x && x < b.x+brickWidth && y > b.y && y < b.y+brickHeight) {
              dy = -dy;
              b.status = 0;
              score++;
              if(score == brickRowCount*brickColumnCount) {
                  alert("YOU WIN, CONGRATS!");
                  document.location.reload();
              }
              }
          }
          }
      }
      }
```

03

ORIGINAL CODE 설명

블럭을 생성하기 위한 부분

열 C와 행 R로 이루어진 2차원 배열 bricks를 만든다.
각 객체에는 벽돌의 위치를 나타낼 x, y가 있다.

배열 안에 있는 벽돌을 반복해서 화면에 그려줄 함수
행, 열 반복을 통해 벽돌을 그리는데, 모든 벽돌의
좌표가 (0, 0)으로 위치해, brickX, brickY 코드의
연산을 통해 x, y값을 계산하여 벽돌을 그린다.

```
41    // 2차원 배열에 백돌을 담음. 배열은 열 c, 행 r로 이루어져있고, 각 객체에는 벽돌의 위치를 나타낼 x, y를 가지고 있다.
42    let bricks = [];
43    for(let c=0; c<brickColumnCount; c++) {
44    bricks[c] = [];
45    for(let r=0; r<brickRowCount; r++) {
46        bricks[c][r] = { x: 0, y: 0, status: 1 };
47    }
48    }
```

```
119   // 배열 안에 벽돌을 반복해서 화면에 그려줄 함수 만드는 코드
120   // 행, 열 반복을 통해 벽돌을 그리는데, 모든 벽돌의 좌표가 (0,0)에 위치해 연산을 통해 x, y값을 계산하는 코드가 들어있다.
      function drawBricks() {
      for(let c=0; c<brickColumnCount; c++) {
          for(let r=0; r<brickRowCount; r++) {
          if(bricks[c][r].status == 1) {
              let brickX = (r*(brickWidth+brickPadding))+brickOffsetLeft;
126           let brickY = (c*(brickHeight+brickPadding))+brickOffsetTop;
              bricks[c][r].x = brickX;
              bricks[c][r].y = brickY;
              ctx.beginPath();
              ctx.rect(brickX, brickY, brickWidth, brickHeight);
              ctx.fillStyle = "#0095DD";
              ctx.fill();
              ctx.closePath();
          }
135       }
136   }
137   }
```

03

ORIGINAL CODE 설명

공을 튕겨내기 위한 부분

공이 벽에 닿았다면 튕겨낼 수 있도록 해야한다.
추가로 바닥에 닿는다면 목숨이 줄어들 수 있게 한다.

```
149     function draw() {
150     // 매 프레임마다 공을 그릴 때 이전 프레임을 지워주지 않으면 공이 흔적을 남기게 된다. 따라서 내용을 지워주기 위해 clearRect()가 필요하다.
151     ctx.clearRect(0, 0, canvas.width, canvas.height);
152     drawBricks();
153     drawBall();
154     drawPaddle();
155     drawScore();
156     drawLives();
157     collisionDetection();
158
159     if(x + dx > canvas.width || x + dx < 0) {
160         dx = -dx;
161     }
162     if(y + dy < 0) {
163         dy = -dy;
        }
        else if(y + dy > canvas.height) {
            if(x > paddleX && x < paddleX + paddleWidth) {
            dy = -dy;
            }
            else {
170         lives--;
171         if(!lives) {
172             alert("GAME OVER");
173             document.location.reload();
174         }
175         else {
176             x = canvas.width/2;
177             y = canvas.height-30;
178             dx = 3;
179             dy = -3;
180             paddleX = (canvas.width-paddleWidth)/2;
181         }
182         }
183     }
```

03

ORIGINAL CODE 설명

패들을 움직이기 위한 부분

공이 바닥에 당기 전에 튕기게 할 수 있는 도구로
패들이 필요하여 패들을 넣고 움직일 수 있게 하였다.

```
185     // paddle을 움직이게 하는 코드
186     if(rightPressed) {
187         paddleX += 7;
188     }
189     else if(leftPressed) {
190         paddleX -= 7;
191     }
192
193     x += dx;
194     y += dy;
195     requestAnimationFrame(draw);
196     }
197
```

04

ORIGINAL CODE의 개선점

ORIGINAL CODE를 실행시켰을 때
수정했으면 좋겠다고 느꼈던 부분입니다.

1ST
제목과 설명, 조작법

2ND
색상 변경

3RD
난이도 설정

4TH
남은 블럭 수

04

ORIGINAL CODE의 개선점

ORIGINAL CODE를 실행시켰을 때
수정했으면 좋겠다고 느꼈던 부분입니다.

5TH
패들 오류 수정

6TH
공 오류 수정

7TH
키보드 제어 삭제

8TH
게임 끝날 시 안내문

05

UPGRADE
CODE 설명 – 1

타이틀 변경

페이지를 열었을 때 보이는 타이틀 이름을 변경하였다.

05

UPGRADE
CODE 설명 – 2

제목 추가

```
13    <body>
14        <div class = "container-fluic p-5 bg-warning text-white text-center">
15            <h1>BLOCK GAME</h1>
16            <p>소프트웨어학과 2020875042 이수현</p>
17        </div>
```

ORIGINAL CODE에서 페이지에 들어갔을 때 게임만 보이는 것이 약간 신경쓰였다.

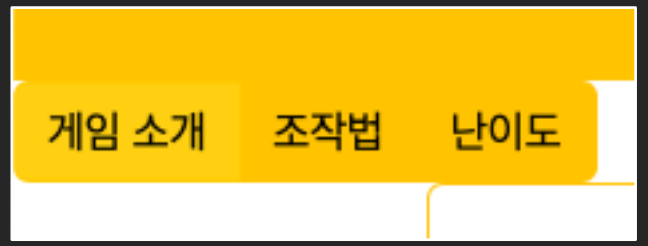
아무것도 없는 화면보다는 제목을 추가하는 것이 좋을 것이라고 판단해, 제목을 넣기로 했다.
그냥 글씨만 보이게 하기에는 밋밋한 것 같아, Bootstrap을 사용하여 스타일을 지정하였다.

05

UPGRADE
CODE 설명 – 3

메뉴 버튼 추가

게임 소개와 조작법, 난이도에 대한 설명을 추가하기
위해서 버튼을 사용하여 메뉴를 만들어주었다.
마찬가지로 Bootstrap을 사용하였다.




```
19      <div class="btn-group" style="display:flec; justify-content: center;" >
20          <a href="./intro.html" target="_blank" class="btn btn-warning active" aria-current="page">게임 소개</a>
21          <a href="./rule.html" target="_blank" class="btn btn-warning">조작법</a>
22          <a href="./level.html" tarket="_blank" class="btn btn-warning">난이도</a>
23        </div>
```

05

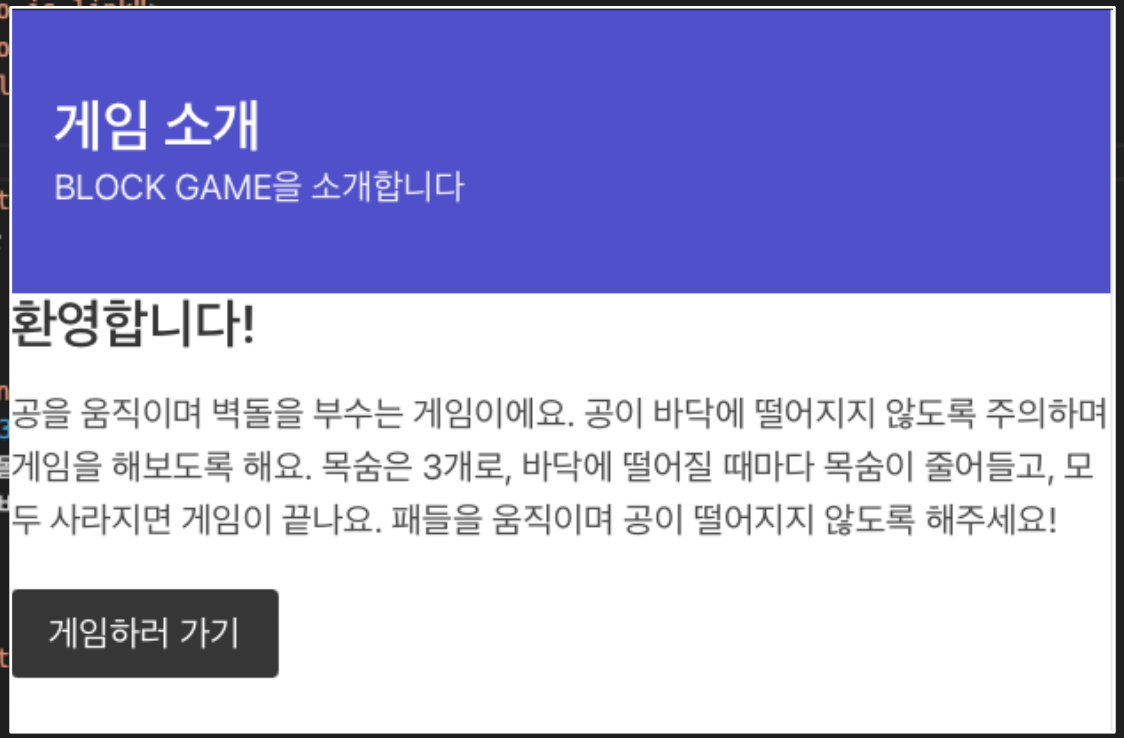
UPGRADE CODE 설명 – 4

게임 소개 추가

게임의 소개가 없는 점이 아쉬웠다.
게임 소개를 위해 새로운 html 파일을 만들어
버튼으로 이동할 수 있게 했다.
게임하러 가기 버튼을 누르면 페이지가 닫힌다.

약간의 디자인이 필요한 것 같아 Bulma를 사용하여
페이지를 만들었다.

```
1   <!DOCTYPE html>
2   <html lang="en">
3   <head>
4       <meta charset="UTF-8">
5       <meta http-equiv="X-UA-Compatible" content="IE=edge">
6       <meta name="viewport" content="width=device-width, initial-scale=1.0">
7       <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bulma@0.9.4/css/bulma.min.css">
8       <title>게임 소개</title>
9   </head>
10  <body>
11      <section class="hero
12          <div class="hero
                <p class="titl
                  게임 소개
                </p>
                <p class="subt
                  BLOCK GAME을
                </p>
              /div>
            ection>
21        <div class="conten
            h3>환영합니다!</h3
            p>공을 움직이며 벽돌
              목숨은 3개로, 바
              패들을 움직이며
26          </p>
27        </div>
28        <button class="but
29  </body>
30  </html>
```



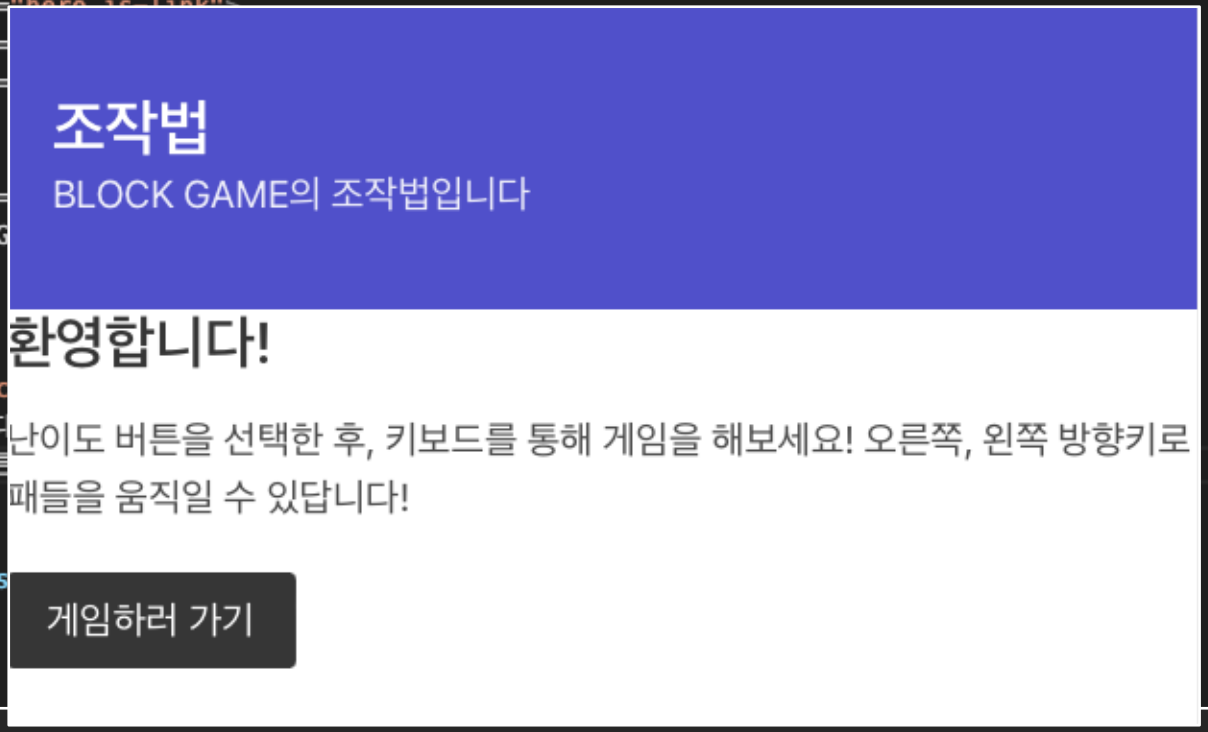
05

UPGRADE CODE 설명 – 5

조작법 추가

간단한 게임이라도 조작법이 필요하다고 생각되었다.
게임 소개와 마찬가지로 새로운 html 파일을 만들어
버튼으로 이동할 수 있게 했다.
게임하러 가기 버튼을 누르면 페이지가 닫힌다.

```
1   <!DOCTYPE html>
2   <html lang="en">
3   <head>
4       <meta charset="UTF-8">
5       <meta http-equiv="X-UA-Compatible" content="IE=edge">
6       <meta name="viewport" content="width=device-width, initial-scale=1.0">
7       <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bulma@0.9.4/css/bulma.min.css">
8       <title>조작법</title>
9   </head>
10  <body>
11      <section class="
12          <div class=
13              <p class=
                조작법
              /p>
              p class=
                BLOCK G
              /p>
              iv>
            tion>
21        <div class="c
22          <h3>환영합니다
23          <p>난이도 버튼
24          </p>
25        </div>
26        <button class
27  </body>
28  </html>
```



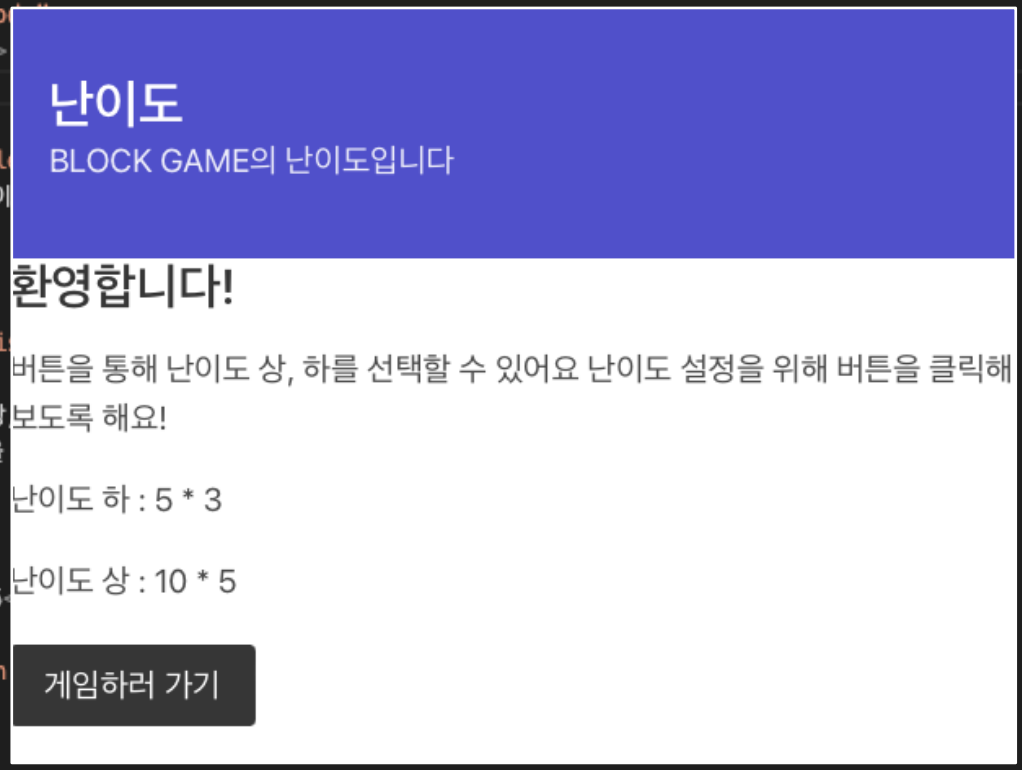
05

UPGRADE CODE 설명 – 6

난이도 설명 추가

난이도를 알 수 있는 페이지를 만들었다.
마찬가지로 새로운 html 파일을 만들어
버튼으로 이동할 수 있게 했다.
게임하러 가기 버튼을 누르면 페이지가 닫힌다.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bulma@0.9.4/css/bulma.min.css">
    <title>난이도</title>
</head>
<body>
    <section class="hero is-link">
        <div class="hero-bo
            <p class="title">
                난이도
            </p>
            <p class="subtitl
                BLOCK GAME의 난이
            </p>
        </div>
    </section>
    <div class="content i
        <h3>환영합니다!</h3>
        <p>버튼을 통해 난이도 상
        난이도 설정을 위해 버튼을
        </p>
        <p>난이도 하 : 5 * 3
        </p>
        <p>난이도 상 : 10 * 5
    </div>
    <button class="button
</body>
</html>
```

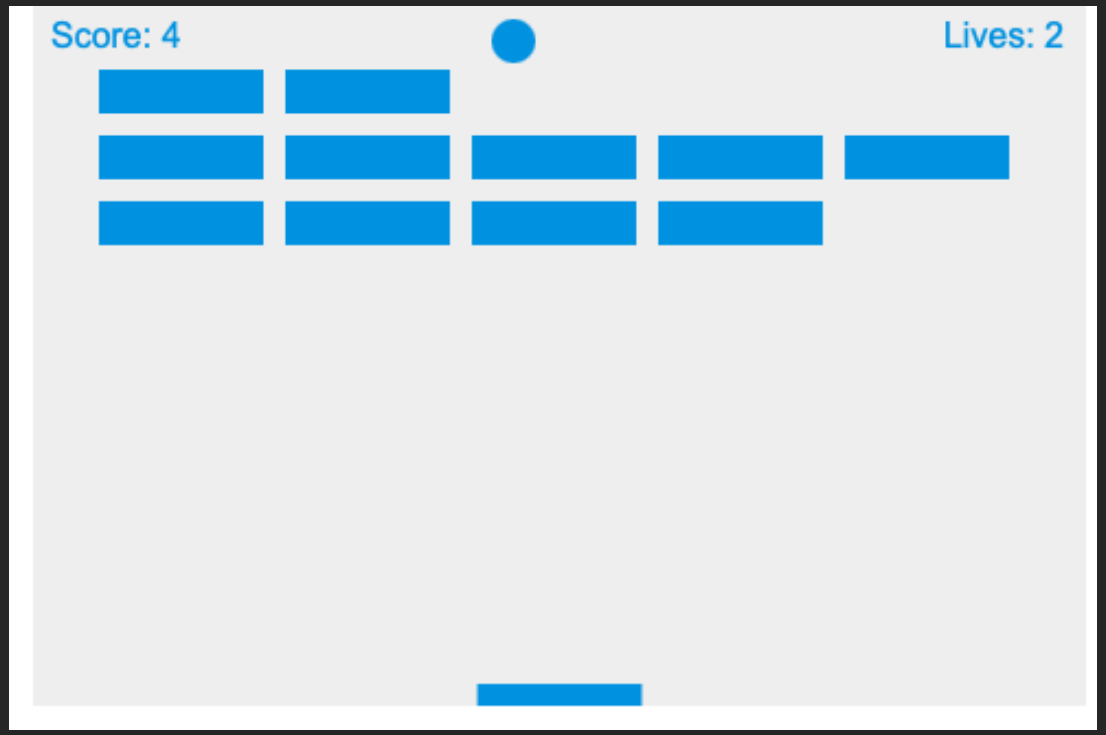


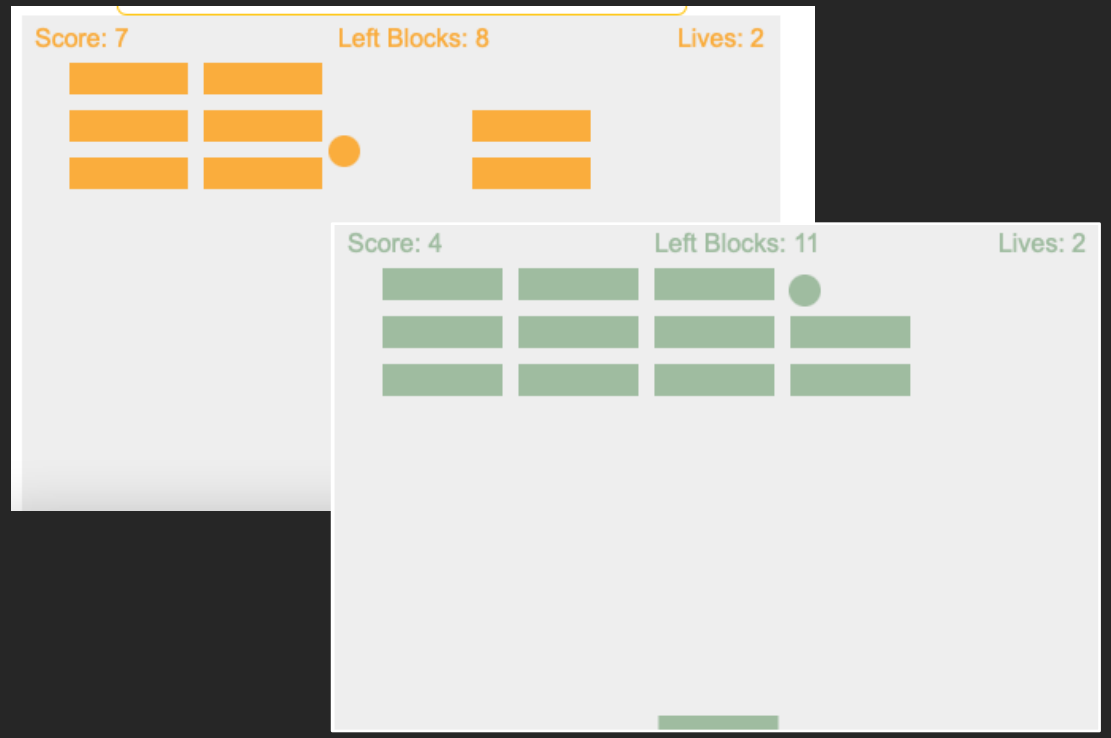
05

UPGRADE CODE 설명 – 7

색상 변경

제목에서 사용했던 Bootstrap의 Warning 색상에 맞춰 변경해주었다.
추가로 난이도가 다르므로, 다른 난이도는 다른 색상으로 보일 수 있게 했다.





05

UPGRADE CODE 설명 – 8

난이도 버튼 추가

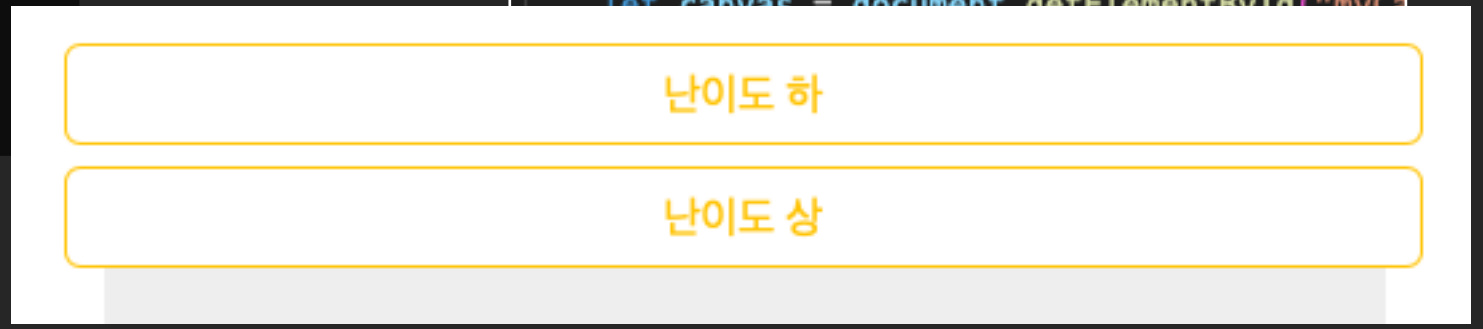
```
40      <div class="d-grid gap-2 col-6 mx-auto">
41          <button type="button" class="btn btn-outline-warning" onclick = "lowlevel()">난이도 하</button>
42          <button type="button" class="btn btn-outline-warning" onclick = "highlevel()">난이도 상</button>
43      </div>
```

난이도를 상, 하로 나누어 게임을 실행할 수 있도록 버튼을 추가했다.

코드는 기존 코드와 동일하지만, 버튼에 따라서 lowlevel() 함수와 highlevel() 함수가 실행되면서 변수의 값이 달라져서 실행된다.
마찬가지로 Bootstrap을 사용하여 디자인하였다.

```
function lowlevel() {
    // canvas 위에 그래픽을 렌더링 하기 위해서 js로 참
    let canvas = document.getElementById("myC
    // 캔버스에 그리기 위해 실질적으로 사용되는 도구인 2D
    let ctx = canvas.getContext("2d");
    let ballRadius = 10;
```

```
function highlevel() {
    // canvas 위에 그래픽을 렌더링 하기 위해서 js로 참조
    let canvas = document.getElementById("myCa
```



05

UPGRADE CODE 설명 – 9

남은 블럭 수 추가

```
375     function leftBlock(){
376         ctx.font = "16px Arial";
377         ctx.fillStyle = "#f0ad4e";
378         ctx.fillText("Left Blocks: "+parseInt(block - score), 200, 20);
379     }
380     function drawScore() {
```

```
263     let leftPressed = false;
264     let brickRowCount = 10;
265     let brickColumnCount = 5;
266     let brickWidth = 35;
267     let brickHeight = 14;
268     let brickPadding = 8;
269     let brickOffsetTop = 30;
270     let brickOffsetLeft = 30;
271     let score = 0;
272     let lives = 3;
```

목숨과 점수만 표시되던 ORIGINAL CODE와 달리
남은 블럭의 수를 알 수 있게 추가했다.

brickRowCount와 brickColumnCount로 총 블럭의 수를 계산해서 block 변수를 선언하고, leftBlock() 함수를 만든다.
block-score을 하니 오류가 생겨 parseInt로 해결해주었다.

05

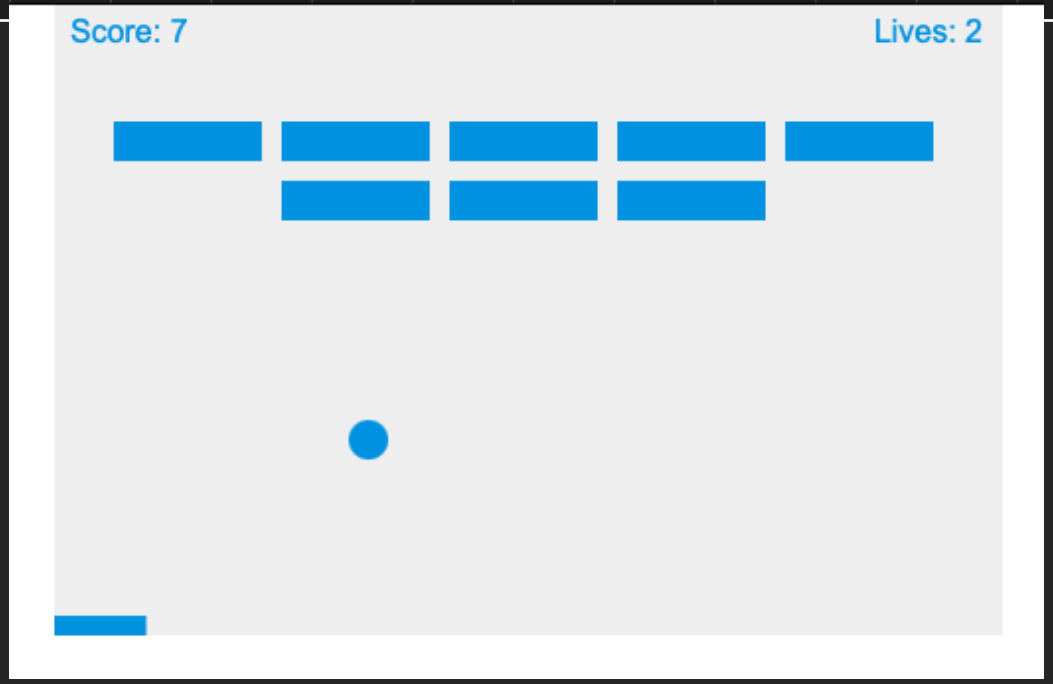
UPGRADE CODE 설명 – 10

패들이 벽에 들어가는 현상 수정

```
226         // paddle을 움직이게 하는 코드
227         // 키를 너무 오래 누르고 있을 때 밖으로 paddle이 사라지는 것을 방지하기 위해 캔버스 안에서만 움직일 수 있게 하는 것
228         if(rightPressed && paddleX < canvas.width-paddleWidth) {
229             paddleX += 7;
230         }
231         else if(leftPressed && paddleX > 0) {
232             paddleX -= 7;
233         }
```

ORIGINAL CODE에서 키보드를 오래 누르고 있을 때, 패들이 벽 안쪽으로 들어가는 현상이 있었다.

캔버스 영역 밖으로 패들이 사라질 수 있어서 생긴 문제이다. 따라서 영역을 캔버스 안으로만 지정해준다. 패들은 캔버스의 왼쪽 끝 0에서부터 오른쪽 끝인 canvas.width-paddleWidth에서 움직일 수 있도록 수정한다.



05

UPGRADE CODE 설명 – 11

공이 벽에 들어가는 현상 수정

ORIGINAL CODE에서 공이 벽에 부딪혔을 때, 공이 벽 안쪽으로 들어가는 것처럼 보였다.

공이 벽의 충돌을 감지할 때, 기준을 공의 원점에 두고 계산해서 생긴 문제점이었다. 따라서 기준을 원의 둘레에 두고 계산하기 위해 공의 원점과 벽 사이의 거리가 공의 반지름과 같아졌을 때, 공이 튕겨 나올 수 있도록 수정하였다.

```
199     // 벽에 충돌했을 때 튕겨나올 수 있게 하는 코드 만약 ballRadius가 아니라면 벽 안쪽으로 약간 들어가게 됨
200     if(x + dx > canvas.width-ballRadius || x + dx < ballRadius) {
201         dx = -dx;
202     }
203     if(y + dy < ballRadius) {
204         dy = -dy;
205     }
206     else if(y + dy > canvas.height-ballRadius) {
207         if(x > paddleX && x < paddleX + paddleWidth) {
208         dy = -dy;
209         }
            else {
            lives--;
            if(!lives) {
                alert("GAME OVER");
                document.location.reload();
            }
            else {
                x = canvas.width/2;
                y = canvas.height-30;
                dx = 3;
                dy = -3;
                paddleX = (canvas.width-paddleWidth)/2;
            }
            }
    }
```

05

UPGRADE
CODE 설명 – 12

마우스 제어 부분 제거

키보드와 마우스로 모두 게임을 진행하다 보니
두개가 함께 작용하는 것이 불편하다고 생각되었다.

마우스로 게임을 제어하던 부분을 주석 처리하여
게임은 키보드만 진행될 수 있도록 하였다.

```
111        // function mouseMoveHandler(e) {
112        // var relativeX = e.clientX - canvas.offsetLeft;
113        // if(relativeX > 0 && relativeX < canvas.width) {
114        //     paddleX = relativeX - paddleWidth/2;
115        // }
116        // }
```

```
// document.addEventListener("mousemove", mouseMoveHandler, false);
```

05

UPGRADE CODE 설명 – 13

게임 마무리 시 안내문 출력

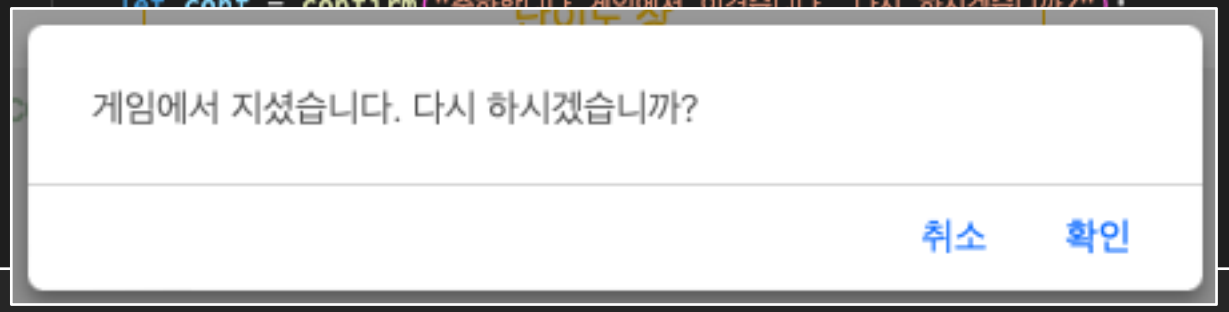
게임 종료 후 자동으로 페이지가 새로고침 되는 것보다 안내문이 필요하다고 생각했다.

alert 대신 confirm을 사용해 다시 한다는 문구에 확인을 누르면 페이지를 reload 하고, 그렇지 않으면 페이지에서 벗어날 수 있도록 만들었다.

```
224 ⌄      if(!lives) {
225 ⌄          // alert("GAME OVER");
226            // document.location.reload();
227            let conf = confirm("게임에서 지셨습니다. 다시 하시겠습니까?");
228 ⌄              if(conf == true) {
229                    document.location.reload();
230                } else if(conf == false) {
231                    window.close();
               }
       }
```

```
       score++;
       if(score == brickRowCount*brickColumnCount) {
           // alert("YOU WIN, CONGRATS!");
           // document.location.reload();

           let conf = confirm("축하합니다. 게임에서 이겼습니다. 다시 하시겠습니까?");
139
140
141
142
143
```



05

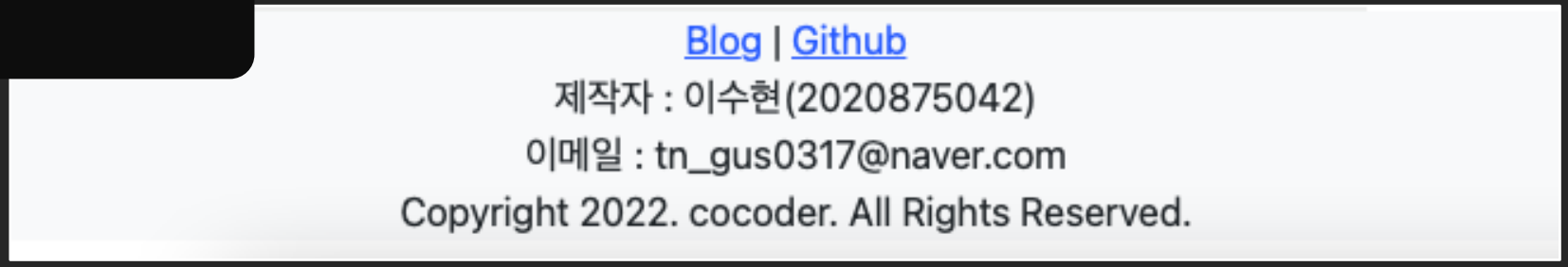
UPGRADE CODE 설명 – 14

FOOTER 추가

제일 아랫부분에 정보를 담을 수 있는 공간을 만들기 위해 footer 태그를 이용했다.

blog는 없어 naver로 연결되게 만들었고, github는 본인의 github 주소로 이어질 수 있도록 넣었다.
제작자와 이메일 주소를 넣었다.

```
473     </script>
474     <footer class = "bg-light text-center text-lg-start">
475         <nav>
476             <a href='https://naver.com' target='_blank'>Blog</a> |
477             <a href='https://github.com/hyeon317' target='_blank'>Github</a>
478         </nav>
479         <p>
480             <span>제작자 : 이수현(2020875042)</span><br/>
481             <span>이메일 : tn_gus0317@naver.com</span><br/>
482             <span>Copyright 2022. cocoder. All Rights Reserved.</span>
483         </p>
484     </footer>
```



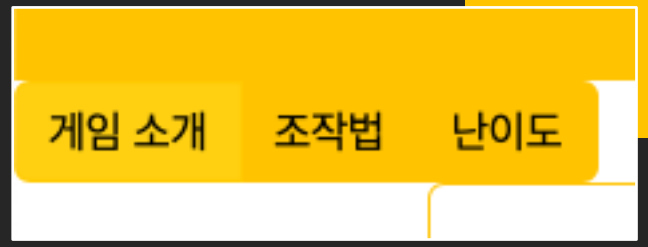
05

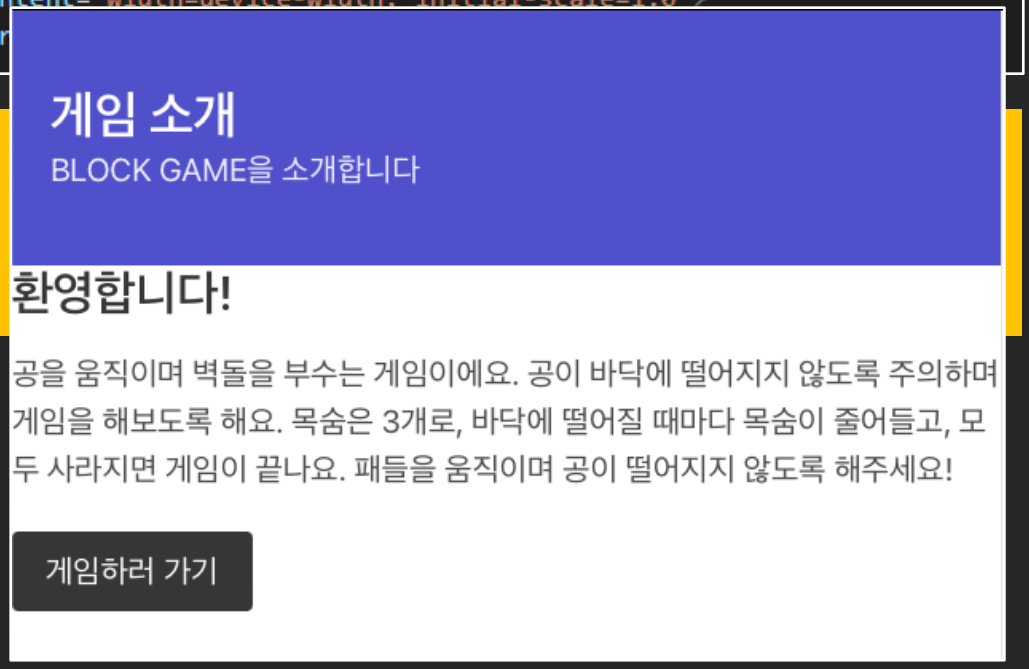
UPGRADE CODE 설명 – 15

framework 사용

앞서 설명에서 나왔던 것처럼
Bootstrap과 Bulma를 이용했다.

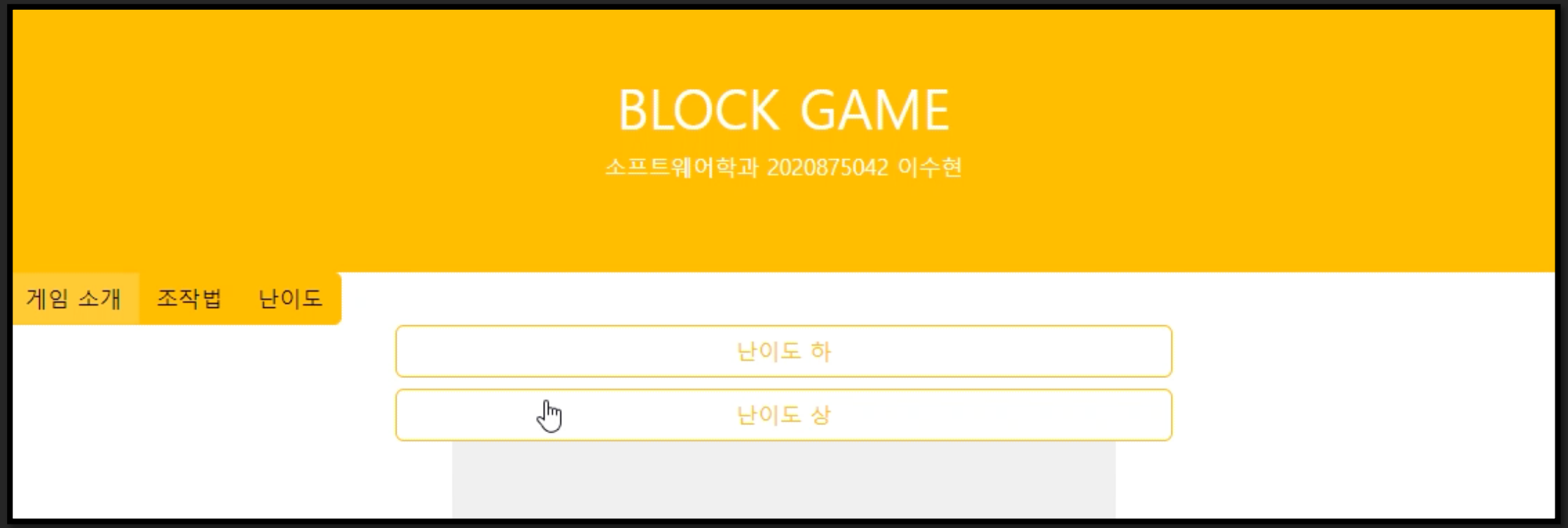
```
3   <head>
4       <meta charset="UTF-8">
5       <meta http-equiv="X-UA-Compatible" content="IE=edge">
6       <meta name="viewport" content="width=device-width, initial-scale=1.0">
7       <link rel="stylesheet" hr
```





06

BLOCK GAME UPGRADE 시연



07

REFERENCE

이번 프로젝트를 하면서 참고한 자료들입니다.

원본 코드	https://developer.mozilla.org/ko/docs/Games/Tutorials/2D_Breakout_game_pure_JavaScript
Bootstrap	https://getbootstrap.com/docs/5.2/layout/containers/
Bulma	https://bulma.io/documentation/elements/button/
난이도버튼	https://spooohome.blogspot.com/2021/05/breakout.html
GITHUB	https://github.com/hyeon317/GameProgramming/tree/main/1116%20html%20project
ORIGINAL _ github.io	https://hyeon317.github.io/original_html.github.io/
UPGRADE _ github.io	https://hyeon317.github.io/html_project.github.io/

THANK YOU

FOR WATCHING