

Abstract geometric lines in the top left corner, consisting of several overlapping, irregular polygons and lines in a light beige color.

CANVAS 게임 만들기

소프트웨어학과 2020975032 안지현

목차

1.원래코드 분석

2.개선한 코드 분석

3.참고자료

Ball게임

Point : 0



코드 분석-1

```
<body>
  <canvas id="canvas" width="500" height="800"></canvas>
  <script>
    const canvas = document.querySelector("#canvas");
    const ctx = canvas.getContext("2d");

    let x = canvas.width / 2;
    let y = canvas.height - 30;

    let mx = 2,
        my = -2;

    let up = false,
        down = false,
        right = false,
        left = false;

    let rw = Math.random() * canvas.width,
        rh = Math.random() * canvas.height;

    let score = 0;
    canvas.style.border = "1px solid #222";

    setInterval(move, 10);
```

querySelector: canvas 호출

setInterval(실행 할 함수, 시간 간격)

코드 분석-2

```
const keyDownHandler = (e) => {  
  if (e.key == "Right" || e.key == "ArrowRight") {  
    right = true;  
  } else if (e.key == "Left" || e.key == "ArrowLeft") {  
    left = true;  
  } else if (e.key == "Up" || e.key == "ArrowUp") {  
    up = true;  
  } else if (e.key == "Down" || e.key == "ArrowDown") {  
    down = true;  
  }  
}
```

```
const keyUpHandler = (e) => {  
  if (e.key == "Right" || e.key == "ArrowRight") {  
    right = false;  
  } else if (e.key == "Left" || e.key == "ArrowLeft") {  
    left = false;  
  } else if (e.key == "Up" || e.key == "ArrowUp") {  
    up = false;  
  } else if (e.key == "Down" || e.key == "ArrowDown") {  
    down = false;  
  }  
}
```

KeyDownHandler: 키를 눌렀을때 동작

KeyUpHandler: 키를 떼었을때 동작

코드 분석-3

```
function targetDraw() {
    ctx.beginPath();
    ctx.arc(rw, rh, 10, 0, Math.PI * 2);
    ctx.fillStyle = "red";
    ctx.fill();
    ctx.closePath();
}

function point() {
    if ((x <= rw + 15 && x >= rw - 15) && (y <= rh + 15 && y >= rh - 15)) {
        score++;
        rw = Math.random() * canvas.width;
        rh = Math.random() * canvas.height;
    }
    ctx.beginPath();
    ctx.font = "italic 22px Calibri";
    ctx.fillStyle = "blue";
    ctx.fillText(`Point : ${score}`, 5, 20);
    ctx.closePath;
}
```

ctx.beginPath(): 새로운 선 그리기

ctx.arc(x,y,반지름,시작각,끝각,회전반시계방향)

BALL GAME => SHOOTING GAME

1. 게임 시작화면 설정
2. 배경화면, 우주선, 몬스터, 총 등의 삽화 삽입
3. 게임 룰의 변화
 - 몬스터가 위에서 아래로 떨어짐
 - 우주선에서 총이 나오도록
 - 총으로 몬스터를 맞추면 +1점
 - 몬스터가 바닥에 닿으면 게임이 종료
4. 총을 맞았을때, 게임이 종료되었을때의 효과음 설정

1. 게임 시작화면 설정

```
<style>
body{
  background: #000000;
}
```

```
#wrapper{
  width:700px;
  height:500px;
  border:5px solid #000;
  margin: auto;
  text-align: center;
}
```

```
#wrapper h1{
  text-align: center;
  color: #FF0000;
  font-weight: bold;
  font-size: 70px;
}
```

```
#container{
  height: 120px;
}
```

```
#wrapper p{
  text-align: center;
  color: #00FFFF;
  font-weight: bold;
  font-size: 35px;
  display: none;
}
```

```
#wrapper button{
  padding: 10px;
}
</style>
```

```
<script>
let flag=true;
function init(){
  document.querySelector("button").addEventListener("click", function(){
    location.href="shooting_game.html";
  });
}
```

```
<!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" />
  <title>Document</title>
</head>
<body>
  <h1>슈팅게임</h1>
  <script src="./main.js"></script>
</body>
</html>
```

```
<p>INSERT COIN</p>
</div>
<button>Game Start</button>
</div>
</body>
</html>
```


2. 배경화면, 우주선, 몬스터, 총 등의 삽화 삽입

```
function loadImage() {
    backgroundImage = new Image();
    backgroundImage.src = "images/backgroundImage.png";

    spaceshipImage = new Image();
    spaceshipImage.src = "images/spaceship.png";

    bulletImage = new Image();
    bulletImage.src = "images/bullet.png";

    enemyImage = new Image();
    enemyImage.src = "images/enemy.png";

    gameOverImage = new Image();
    gameOverImage.src = "images/gameover.png";
}

let keysDown = {};
function setupKeyboardListener() {
    // 키를 눌렀을때(down)
    document.addEventListener("keydown", function (e) {
        keysDown[e.keyCode] = true;
    });
    // 키 (누튼거) 손때었을때(up)
    document.addEventListener("keyup", function (e) {
        delete keysDown[e.keyCode];

        if (e.keyCode == 32) {
            createBullet(); // 총알 생성하는 함수
        }
    });
}
```

3_1. 몬스터가 위에서 아래로 떨어짐

```
let enemyList = [];  
function Enemy() {  
  this.x = 0;  
  this.y = 0;  
  this.init = () => {  
    this.y = 0;  
    this.x = RandomValue(0, canvas.width - 32);  
  
    enemyList.push(this);  
  };  
  this.update = function () {  
    this.y += 1.7; // 적군의 속도 조절  
  };  
}
```

```
function render() {  
  ctx.drawImage(backgroundImage, 0, 0, canvas.width, canvas.height);  
  ctx.drawImage(spaceshipImage, spaceshipX, spaceshipY);  
  ctx.fillText(`Score:${score}`, 20, 30); // 가, 세  
  ctx.fillStyle = "white";  
  ctx.font = "20px Arial";  
  for (let i = 0; i < bulletList.length; i++) {  
    if (bulletList[i].alive) {  
      ctx.drawImage(bulletImage, bulletList[i].x, bulletList[i].y);  
    }  
  }  
  
  for (let i = 0; i < enemyList.length; i++) {  
    ctx.drawImage(enemyImage, enemyList[i].x, enemyList[i].y);  
  }  
}
```

```
function createEnemy() {  
  const interval = setInterval(function () {  
    let e = new Enemy();  
    e.init();  
  }, 1500); // (호출하고싶은 함수, 시간마다-ms) , 1초 =  
}
```

3_2. 우주선에서 총이 나오도록

```
function Bullet() {
  this.x = 0;
  this.y = 0;
  this.init = () => {
    this.x = spaceshipX + 5;
    this.y = spaceshipY - 60;
    this.alive = true; //true면 살아있는 총알, false면 죽은 총알
  };
  bulletList.push(this);
  this.update = function () {
    this.y -= 7;
  };

  this.checkHit = () => {
    /* 총알.y <= 적군.y and
    총알.x >= 적군.x and 총알.x <= 적군.x+너비*/
    for (let i = 0; i < enemyList.length; i++) {
      if (
        this.y <= enemyList[i].y &&
        this.x >= enemyList[i].x-15 &&
        this.x <= enemyList[i].x + 15
      ) {
        //총알이 죽게됨, 적군이 없어짐, 점수획득
        audio.play();
        score++;
        this.alive = false; //죽은 총알
        enemyList.splice(i, 1); // i번째에 있는거 하나를 잘라냄.
      }
    }
  };
}
```

3_3. 총으로 몬스터를 맞추면 +1점

```
✓ this.checkHit = () => {  
  ✓ /* 총알.y <= 적군.y and  
    총알.x >= 적군.x and 총알.x <= 적군.x+너비*/  
  ✓ for (let i = 0; i < enemyList.length; i++) {  
    if (  
      this.y <= enemyList[i].y &&  
      this.x >= enemyList[i].x-15 &&  
      this.x <= enemyList[i].x + 15  
    ) {  
      //총알이 죽게됨, 적군이 없어짐, 점수획득  
      audio.play();  
      score++;  
      this.alive = false; //죽은 총알  
      enemyList.splice(i, 1); // i번째에 있는거 하나를 잘라냄.  
    }  
  }  
};  
}
```

3_4. 몬스터가 바닥에 닿으면 게임이 종료

```
let enemyList = [];  
function Enemy() {  
  this.x = 0;  
  this.y = 0;  
  this.init = () => {  
    this.y = 0;  
    this.x = RandomValue(0, canvas.width - 32); // 최대, 최소 받음  
  }  
  enemyList.push(this);  
};  
this.update = function () {  
  this.y += 1.7; // 적군의 속도 조절  
  if (this.y >= canvas.height - 32) {  
    gameOver = true;  
    audio2.play();  
  }  
};  
}
```

4. 몬스터를 맞췄을때, 게임이 종료되었을때의 효과음 설정

```
let backgroundImage, spaceshipImage, bulletImage, enemyImage, gameOverImage;  
let gameOver = false; // true면 게임이 끝남, false면 게임이 안끝남.  
var audio2 = new Audio("fail.mp3");  
let score = 0;
```

// 우주선 좌표

```
let spaceshipX = canvas.width / 2 - 30; // 우주선 높이, 넓이 : 30  
let spaceshipY = canvas.height - 60;
```

```
let bulletList = []; // 총알들을 저장하는 리스트  
var audio = new Audio("Gun_sound.mp3");
```

```
for (let i = 0; i < enemyList.length; i++) {  
  if (  
    this.y <= enemyList[i].y &&  
    this.x >= enemyList[i].x - 15 &&  
    this.x <= enemyList[i].x + 15  
  ) {  
    //총알이 죽게됨, 적군이 없어짐, 점수획득  
    audio.play();  
    score++;  
    this.alive = false; // 죽은 총알  
    enemyList.splice(i, 1); // i번째에 있는거 하나를 잘라냄.  
  }  
}  
;
```

SHOOTING GAME 영상

Space Adventure

Game Start

참고 자료

Ball게임 : <https://blog.naver.com/bdgom73/222669980351>

게임 시작화면 : <https://dw3232.tistory.com/31>

Canvas : <https://lifere.tistory.com/entry/HTML-canvas-%ED%83%9C%EA%B7%B8-Javascript%EB%A1%9C-%EC%A0%9C%EC%96%B4%ED%95%98%EA%B8%B0-%EA%B2%8C%EC%9E%84-%EB%A7%8C%EB%93%A4%EA%B8%B0>

A series of thin, light brown lines of varying lengths and orientations intersecting to form an abstract, geometric pattern on the left side of the slide.

감사합니다.

Github : <https://github.com/JihyeonAn/game/tree/main/1115>