

휴먼 컴퓨터 인터페이스

과제 #1 HTML Canvas

App화면 모사하기



광운대학교
KwangWoon University

2017203020

응용소프트웨어학부 김보섭

목차

- 개요

- 본문

- 논의



광운대학교
KwangWoon University

개요

1. 요구조건과 제약조건 만족도 및 구현사항 요약(표)

요구조건	
하나의 *.html 파일	O
Canvas 요구조건	O
Canvas 그리기	O
JavaScript 이벤트 핸들러	O
JavaScript 객체지향 프로그래밍	O
제약조건	
클라이언트 측 스크립트만 사용 X	O
외부 라이브러리 및 리소스 사용 X	O
구글 크롬 웹 브라우저 호환 필수	O

자세한 구현내용은 본문에서 설명하겠습니다.

광운대학교
KwangWoon University

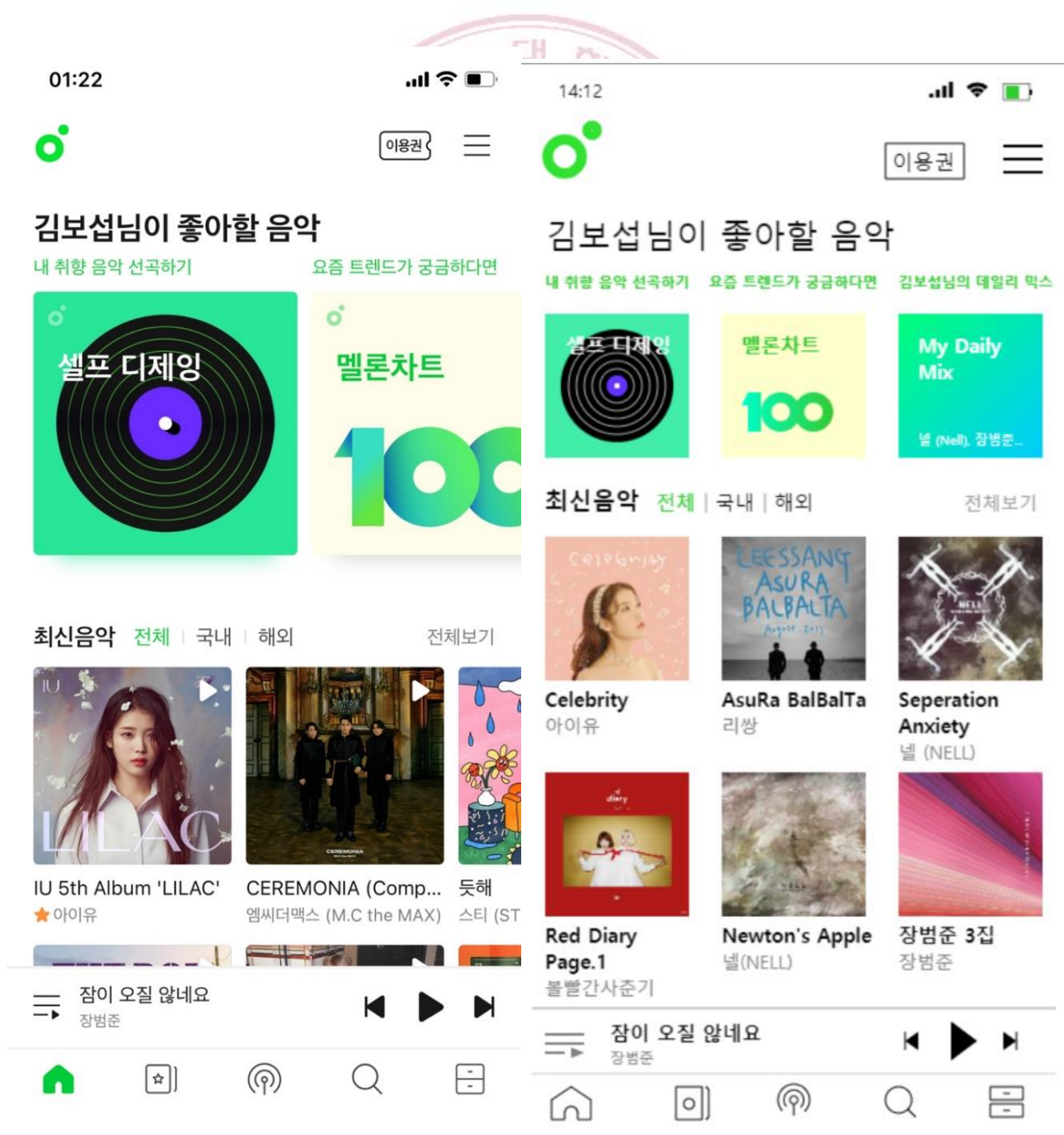
2. 앱에서 선택된 3 개의 화면과 본인이 모사한 3 개의 화면의
병행 제시(그림)

화면 1 (메인화면)

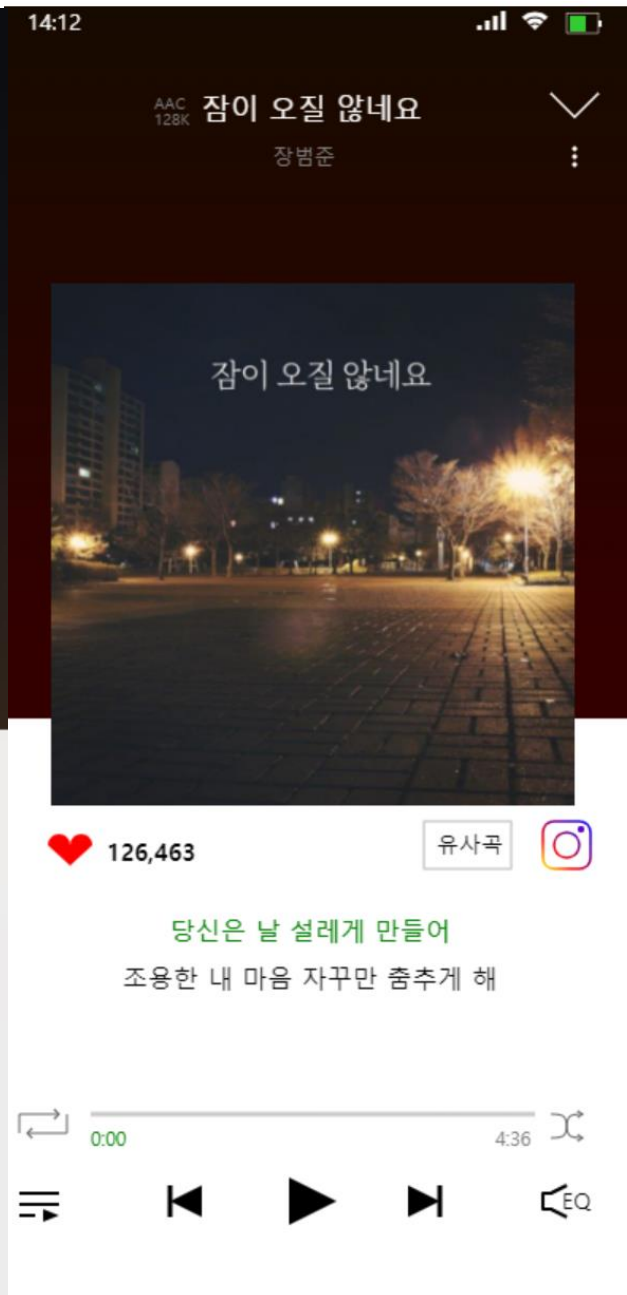
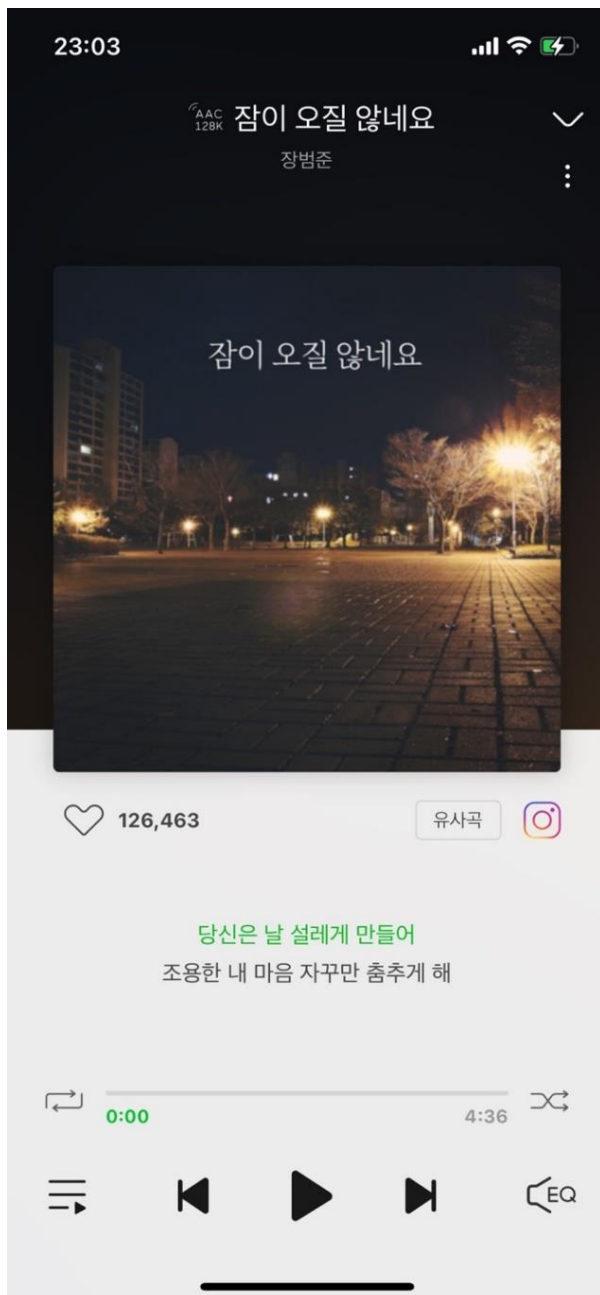
왼쪽의 사진이 앱, 오른쪽의 사진이 모사한 화면

앱 내 화면

모사한 화면



화면 2 (현재 재생중인 음악)





화면 3(나만의 음악서랍)

13:27


LTE

나만의 음악서랍



최근 들은

전체보기



Roller Coaster 외
99곡

♡ 좋아요한

374 >

🎵 내 플레이리스트

19 >

🎯 많이들은

0 >


☆ 팬맷은






26 >

📊 뮤직DNA




>

☰ 잠이 오질 않네요
장범준







14:12




나만의 음악서랍




최근 들은

자주 들은

전체보기



Roller Coaster 외
99곡



폭풍속으로 외
99곡

♡ 좋아요한

374 >

🎵 내 플레이리스트

19 >

🎯 많이들은

0 >

☆ 팬맷은

26 >


📊 뮤직DNA






>

📁 다운로드한

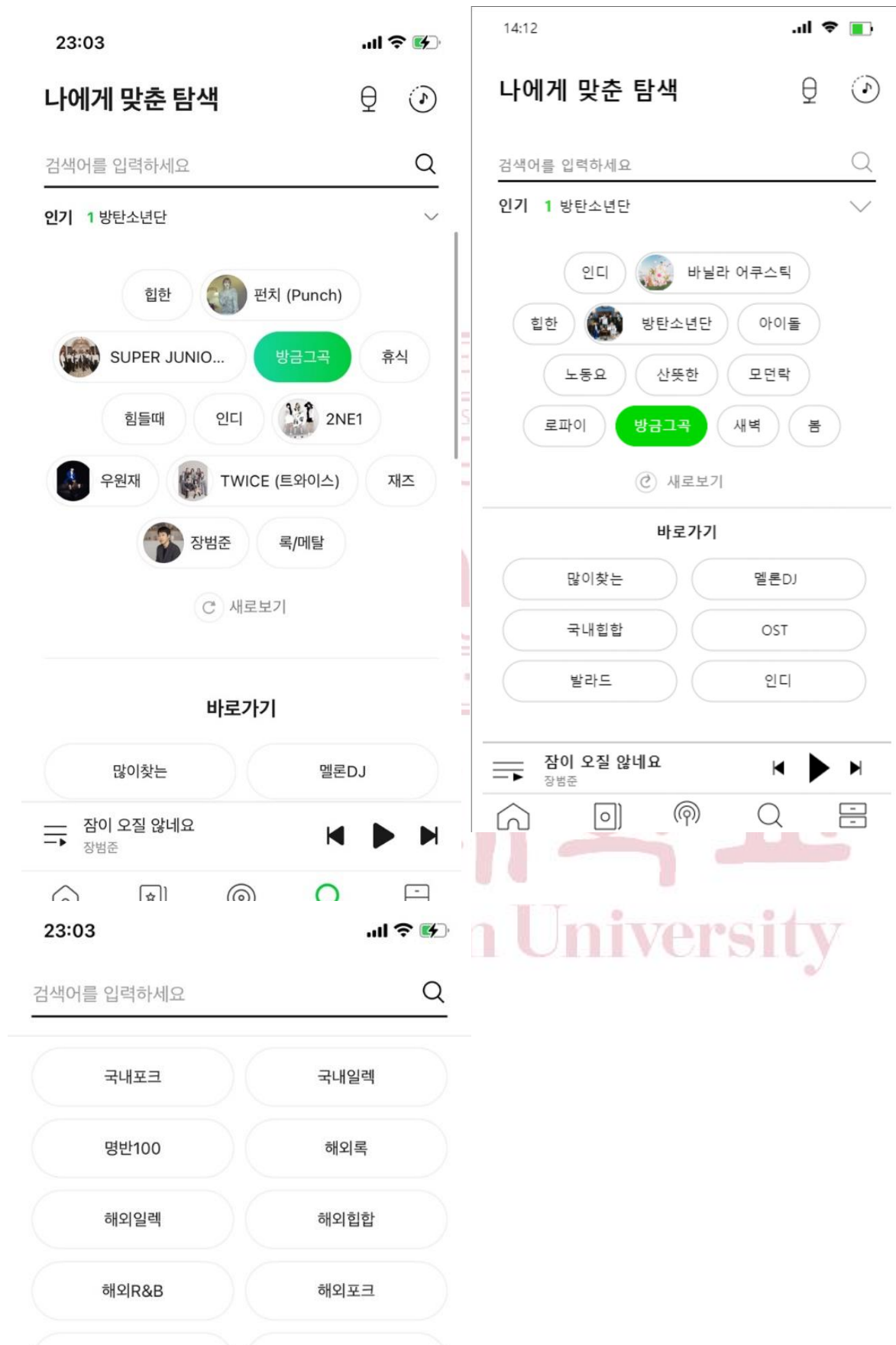
>

☰ 잠이 오질 않네요
장범준

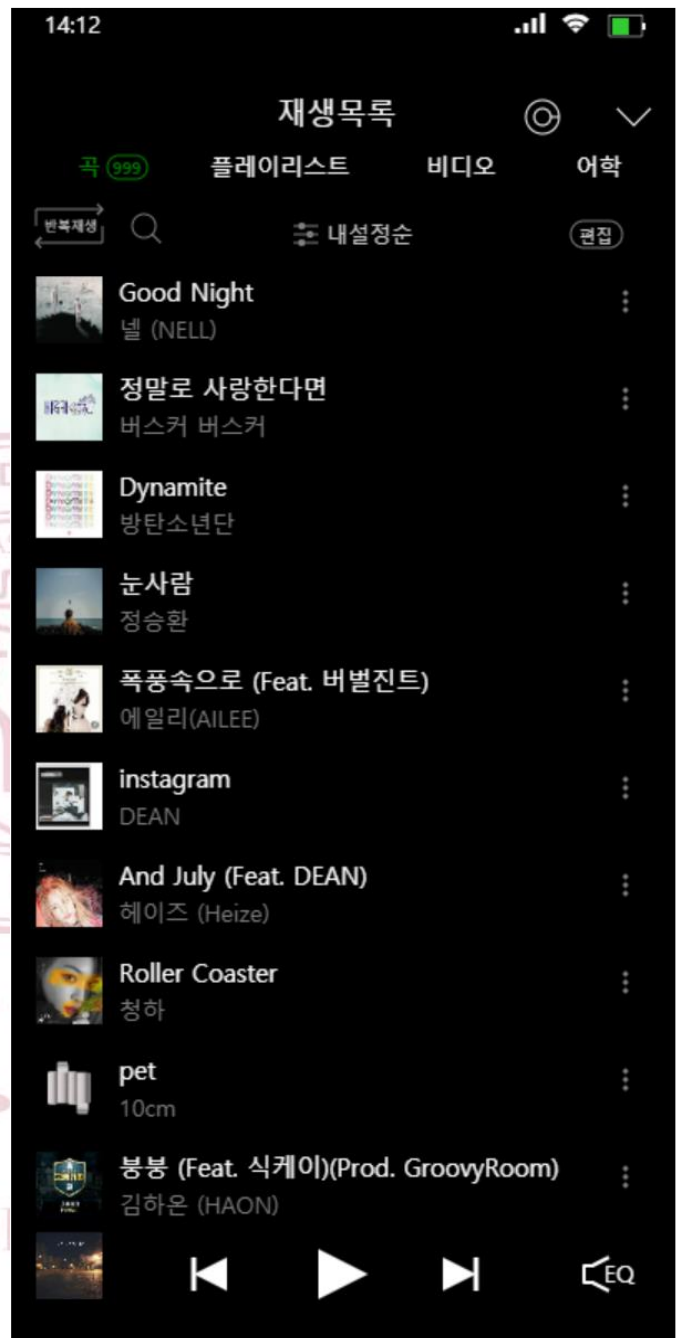
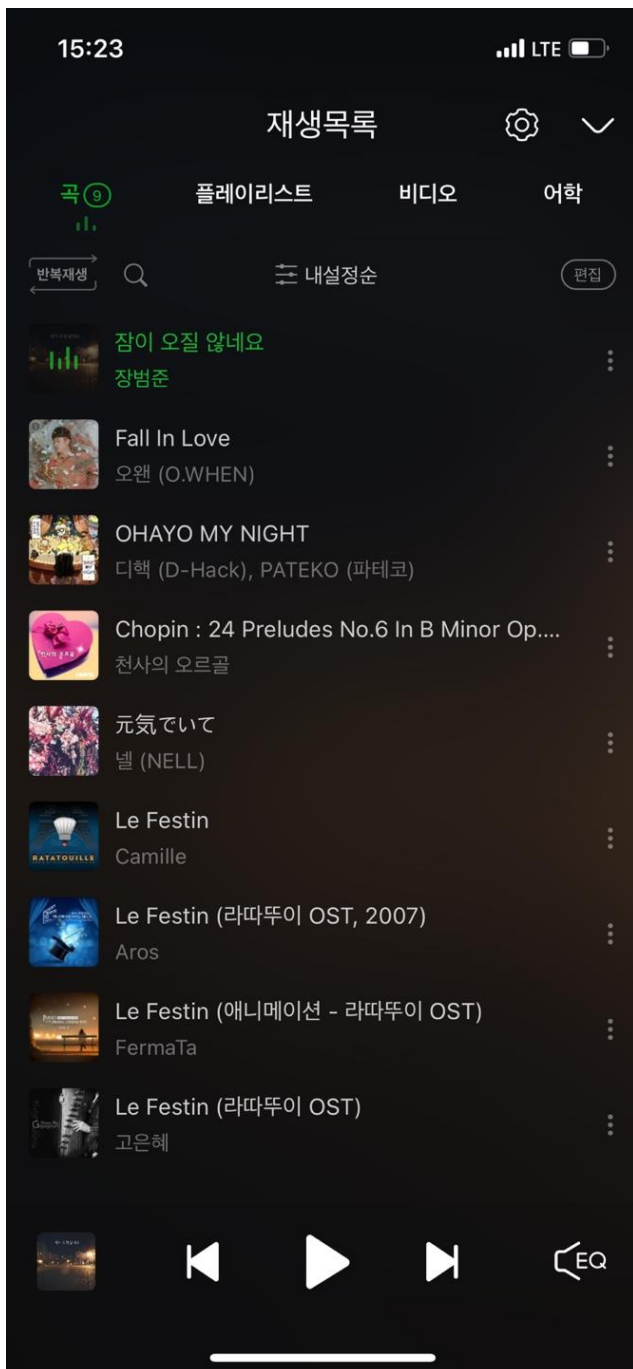




화면 4(검색화면)



화면 5 (재생목록)



음악 어플의 특성상 모사한 화면의 음악은 본래의 화면이 아닌 저의 취향대로 넣어서 모사하였습니다.

본문

소개

앱의 특징 : 멜론은 음원 스트리밍 어플로 국내외의 다양한 음원을 들을 수 있으며, 재생목록 만들기, 본인이 들은 음악을 기준으로 좋아할 만한 노래 추천, 노래 검색 등의 서비스를 제공하는 어플로 초록색의 대표 색상과 깔끔한 UI가 전반적인 어플의 특징입니다

앱을 선택한 이유 : 제가 음악 감상을 좋아하여 다른 어플보다 익숙한 음악 어플로 선정하였으며 여러 음악 어플들 중에서 가장 많이 사용했었고, 깔끔한 UI가 매력적으로 다가와서 이 어플로 선정하게 되었습니다.

전반적인 모사 방안 : 앱의 UI가 전반적으로 간단한 도형으로 이루어져 있고, 비교적 복잡한 모양의 도형은 적기 때문에 단순화 시키지 않고 최대한 유사하게 모사하려고 노력하였습니다. 그리고 위에서 언급한 것처럼, 음악 어플이니 만큼 굳이 캡처 했을 때의 화면을 그대로 모사하는 것 보다는 제가 좋아하는 노래와 노래의 앨범으로 어플의 화면을 모사하였습니다.

캔버스 그리기

전체적인 HTML, CSS, JavaScript 코드 구성

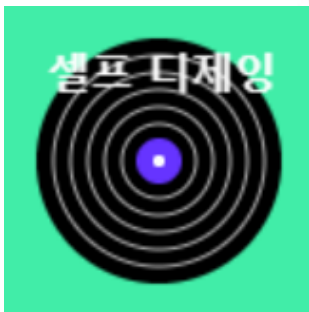
우선 HTML 에서는 전체적인 문서의 뼈대를 구성하고 있습니다
body 에서 canvas 를 이용하고 있습니다

CSS 에서는 간단하게 canvas 의 배경의 style 을 지정하고
있습니다

JavaScript 부분은 저의 코드의 주요 부분이며, 각 화면별로
별개의 함수를 만든 다음 클릭을 하면 다음 화면을 호출하는
방식으로 구성하였습니다

각각의 캔버스 요소를 그리기 위한 방법과 그 결과 예시
이미지

멜론은 대체로 UI 가 깔끔한 어플이기 때문에 Canvas 로 구현하는데 비교적 용이하였습니다.



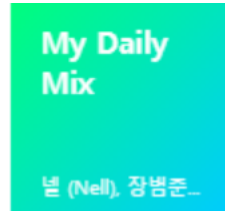
```
context.fillStyle = 'rgba(65, 237, 168, 1)';
context.fillRect(10, 180, 110, 110);

context.beginPath();
context.fillStyle = 'rgba(0, 0, 0, 1)';
context.arc(65, 235, 43, 0, Math.PI * 2);
context.fill();
let j = 5;
while(j > 0){
  context.beginPath();
  context.strokeStyle = 'rgba(255, 255, 255, 1)';
  context.lineWidth = 0.7;
  context.arc(65, 235, 43 - 6 * j, 0, Math.PI * 2);
  context.stroke();
  j -= 1;
}
context.beginPath();
context.fillStyle = 'rgba(102, 52, 255, 1)';
context.arc(65, 235, 8, 0, Math.PI * 2);
context.fill();

context.beginPath();
context.fillStyle = 'rgba(255, 255, 255, 1)';
context.arc(65, 235, 2, 0, Math.PI * 2);
context.fill();

context.font='bold 15px malgum gothic';
context.fillStyle = 'rgba(255, 255, 255, 1)';
context.fillText('셀프 디자인', 26, 210);//여기!!
```

메인페이지의 셀프디제잉 아이콘입니다. 우선 이 부분은 반복문을 이용하여 원을 계속 그려서 디스크의 모양을 구현하였습니다.



```
context.fillStyle = 'rgba(255, 255, 204, 1)';
context.fillRect(145, 180, 110, 110);

context.font='bold 15px malgum gothic';
context.fillStyle = 'rgba(51, 204, 51,1)';
context.fillText('멜론차트', 160, 210);

var green_mint = context.createLinearGradient(160, 0, 230, 0);
green_mint.addColorStop(0, 'rgba(0, 255, 204, 1)');
green_mint.addColorStop(1, 'rgba(51, 204, 51, 1)');
context.fillStyle = green_mint;
context.beginPath();
context.moveTo(160, 250);
context.lineTo(165, 240);
context.lineTo(175, 240);
context.lineTo(175, 270);
context.lineTo(165, 270);
context.lineTo(165, 250);
context.lineTo(160, 250);
context.fill();

context.beginPath();
context.arc(190, 255, 15, 0, Math.PI * 2, false);
context.fill();

context.beginPath();
context.arc(215, 255, 15, 0, Math.PI * 2, false);
context.fill();

context.fillStyle = 'rgba(255, 255, 204, 1)';
context.beginPath();
context.arc(190, 255, 7, 0, Math.PI * 2, false);
context.fill();

context.beginPath();
context.arc(215, 255, 7, 0, Math.PI * 2, false);
context.fill();
//-----
var blue_green = context.createLinearGradient(280, 180, 390, 290);
blue_green.addColorStop(0, 'rgba(0, 255, 123, 1)');
blue_green.addColorStop(1, 'rgba(0, 204, 255, 1)');
context.fillStyle = blue_green;
context.fillRect(280, 180, 110, 110);

context.font='bold 15px malgum gothic';
context.fillStyle = 'rgba(255, 255, 255,1)';
context.fillText('My Daily', 295, 210);
context.fillText('Mix', 295, 230);
context.font='10px malgum gothic';
context.fillText('넬 (Nell), 장범준...', 295, 280);
```

다음 두 아이콘은 그래데이션을 fillStyle 로 넣어서 구현하였습니다. 100 의 숫자는 lineTo 로 다 그린다음 그래데이션으로 채우고 배경색으로 다시 속을 채워서 100 이란 숫자를 완성하였습니다.



Celebrity
아이유



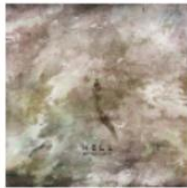
AsuRa BalBalTa
리쌍



Seperation Anxiety
넬 (NELL)



Red Diary Page.1
볼빨간사춘기



Newton's Apple
넬 (NELL)



장범준 3집
장범준

```
var music = [];
function AlbumList(src, aName_1, aName_2, artist, i, j){
    music[i * 3 + j] = new Image();
    music[i * 3 + j].src = src;
    music[i * 3 + j].onload = function(e) {
        context.drawImage(music[i * 3 + j], 10 + 135 * j, 350 + i * 180, 110, 110);
    };
    if(aName_2 != ''){
        context.font='bold 15px malgum gothic';
        context.fillStyle = 'black';
        context.fillText(aName_1, 10 + 135 * j, 480 + i * 180);
        context.fillText(aName_2, 10 + 135 * j, 500 + i * 180);

        context.font='14px malgum gothic';
        context.fillStyle = 'gray';
        context.fillText(artist, 10 + 135 * j, 520 + i * 180);
    }
    else{
        context.font='bold 15px malgum gothic';
        context.fillStyle = 'black';
        context.fillText(aName_1, 10 + 135 * j, 480 + i * 180);

        context.font='14px malgum gothic';
        context.fillStyle = 'gray';
        context.fillText(artist, 10 + 135 * j, 500 + i * 180);
    }
}
AlbumList('./2017203020/AlbumCover/Celebrity.jpg', 'Celebrity', '', '아이유', 0, 0);
AlbumList('./2017203020/AlbumCover/asura_balbalta.jpg', 'AsuRa BalBalTa', '', '리쌍', 0, 1);
AlbumList('./2017203020/AlbumCover/Seperation_Anxiety.jpg', 'Seperation', 'Anxiety', '넬 (NELL)', 0, 2);
AlbumList('./2017203020/AlbumCover/RedDiaryPage1.jpg', 'Red Diary', 'Page.1', '볼빨간사춘기', 1, 0);
AlbumList('./2017203020/AlbumCover/Newtons_apple.jpg', 'Newton\'s Apple', '', '넬 (NELL)', 1, 1);
AlbumList('./2017203020/AlbumCover/JangBeomJune_3rd.jpg', '장범준 3집', '', '장범준', 1, 2);
```

위의 앨범 이미지들을 로딩하기 위해 2 차원 배열을 만들고 함수에서 이 배열을 이용해 이미지들을 로딩하였습니다. 이미지의 경로, 가수이름, 앨범이름, 그리고 위치 등의 인자를 넣어주면 최대 6 개까지 원하는 이미지를 가수이름과 앨범이름, 위치에 생성하게 만들었습니다

i 는 Y 축 좌표이며, j 는 x 축 좌표입니다, 함수에서는 i, j 값에 따라 적절한 값을 더하여 위치를 정하고, 배열에 집어넣어서 Image 객체를 생성하고 있습니다.

또한 위의 Separation Anxiety 와 같이 앨범의 이름이 두 줄을 차지하는 앨범이 있고, Celebrity 와 같이 한 줄만 차지하는 앨범이 존재합니다.

그래서 따로 처리를 하지 않으면 첫 줄만 차지하는 앨범의 이름 바로 밑에 가수 이름이 나오지 않아서 통일성을 주기 어려웠습니다. 그래서 앨범의 이름이 차지하는 길이에 따라 가수명이 바로 밑으로 올 수 있도록 조건문을 달아서 처리하였습니다

처음에는 이 부분을 객체지향 프로그래밍의 요구조건을 만족시키려고 시도하였으나, 저의 코딩실력 부족으로 이미지가 로딩이 되지 않아서 다른 부분에서 구현하였습니다.



126,463

유사곡



당신은 날 설레게 만들어

조용한 내 마음 자꾸만 춤추게 해



```

var endPoints = [{x:350, y:700}, {x:370, y:715}],
    controlPoints = [{x:370, y:700}, {x:350, y:715}];

context.lineWidth = 1;
context.strokeStyle = 'rgba(102, 102, 102, 1)';
context.beginPath();
context.moveTo(endPoints[0].x, endPoints[0].y);
context.bezierCurveTo(controlPoints[0].x, controlPoints[0].y,
    controlPoints[1].x, controlPoints[1].y,
    endPoints[1].x, endPoints[1].y);
context.stroke();

context.beginPath();
context.moveTo(controlPoints[0].x, controlPoints[0].y);
context.bezierCurveTo(endPoints[0].x, endPoints[0].y,
    endPoints[1].x, endPoints[1].y,
    controlPoints[1].x, controlPoints[1].y);
context.stroke();

context.beginPath();
context.moveTo(367, 713);
context.lineTo(370, 715);
context.lineTo(367, 717);
context.stroke();

context.beginPath();
context.moveTo(367, 698);
context.lineTo(370, 700);
context.lineTo(367, 702);
context.stroke(); //shuffle button
//-----
context.beginPath()
context.moveTo(15, 712);
context.lineTo(15, 700);
context.lineTo(40, 700);
context.stroke();

context.beginPath();
context.moveTo(37, 697);
context.lineTo(40, 700);
context.lineTo(37, 703);
context.stroke();

context.beginPath()
context.moveTo(45, 700);
context.lineTo(45, 712);
context.lineTo(20, 712);
context.stroke();

context.beginPath();
context.moveTo(23, 709);
context.lineTo(20, 712);
context.lineTo(23, 715);

```

논의

```

context.fillStyle = 'red';
context.beginPath();
context.arc(40, 530, 7, Math.PI, Math.PI * 2, false);
context.arc(54, 530, 7, Math.PI, Math.PI * 2, false);
context.lineTo(47, 543);
context.lineTo(33, 530);
context.fill();//heart

context.fillStyle = 'black';
context.font = 'bold 15px malgun gothic';
context.fillText('126,463', 70, 540);

function rightButtons(){
    context.strokeStyle = 'rgba(204, 204, 204, 1)';
    context.lineJoin = 'round';
    context.strokeRect(270, 515, 55, 30);

    context.font = '14px malgun gothic';
    context.fillText('유사곡', 278, 535);
    //similar musics
    var insta_grad = context.createLinearGradient(345, 545, 375, 515);
    insta_grad.addColorStop(0, 'yellow');
    insta_grad.addColorStop(0.3, 'red')
    insta_grad.addColorStop(0.6, 'purple');
    insta_grad.addColorStop(1, 'blue');

    context.strokeStyle = insta_grad;
    context.lineJoin = 'round';
    context.lineWidth = 1.5;
    context.beginPath();
    context.arc(352, 522, 7, Math.PI, Math.PI * 3/2, false);
    context.lineTo(368, 515);
    context.arc(368, 522, 7, Math.PI * 3/2, 0, false);
    context.lineTo(375, 538);
    context.arc(368, 538, 7, 0, Math.PI * 1/2, false);
    context.lineTo(352, 545);
    context.arc(352, 538, 7, Math.PI * 1/2, Math.PI, false);
    context.lineTo(345, 522);
    context.stroke();

    context.beginPath();
    context.arc(360, 530, 8, 0, Math.PI * 2, false);
    context.stroke();







    context.beginPath();
    context.fillStyle = 'rgba(103, 0, 204, 1)';
    context.arc(369, 520, 2, 0, Math.PI * 2, false);
    context.fill(); // instagram button

```

위 부분은 재생중인 음악 화면에서 중요 부분입니다.

우선 빨간색 하트는 두개의 반원과 직선으로 만들었고, 인스타그램 아이콘은 여러가지 원색의 색을 이용하여 그라데이션을 이용하였고, 아이콘 특유의 모서리가 둥근 느낌을 만들기 위해 lineTo 로 직선을 그리다가 모서리부분은 사반원을 이용하여 아이콘을 만들었습니다. 물론 round

우하단의 무작위 재생 버튼은 배지어 곡선 2 개를 이용하여 무작위재생 버튼 특유의 교차점을 지나는 두개의 곡선을 구현하였습니다. 첫번째 배지어 곡선과 두번째 배지어 곡선이 서로 대칭적인 모습을 이루게 하기 위해 첫번째 배지어 곡선의 end point 는 두번째 배지어 곡선의 control point 이며, 두번째 곡선의 end point 역시 첫번째 곡선의 control point 입니다

-  좋아요한
-  내 플레이리스트
-  많이들은
-  팬맷은
-  뮤직DNA
-  다운로드한

```

context.beginPath();
context.moveTo(15, 535);
context.lineTo(22, 535);
context.lineTo(25, 528); //first angle
context.lineTo(17, 548);
context.lineTo(35, 535); //second angle
context.lineTo(30, 541);
context.lineTo(33, 548); //third angle
context.lineTo(25, 543);
context.lineTo(17, 548); //forth angle
context.lineTo(20, 541);
context.lineTo(15, 535); //back to first point
context.stroke();

context.fillText("팬택은", 55, 545);
//-----
context.strokeRect(15, 595, 22, 22);
context.fillText("뮤직DNA", 55, 610);
i = 0;
while(i < 3){
    context.beginPath();
    context.moveTo(22 + i * 4, 612);
    context.lineTo(22 + i * 4, 608 - i*3);
    context.stroke();
    i++;
}
//-----
context.beginPath();
context.moveTo(23, 673);
context.lineTo(15, 673);
context.lineTo(15, 658);
context.lineTo(23, 658);
context.lineTo(24, 661);
context.lineTo(36, 661);
context.lineTo(36, 668);
context.stroke();

context.beginPath();
context.moveTo(27, 671);
context.lineTo(31, 674);
context.lineTo(35, 671);
context.stroke();

context.beginPath();
context.moveTo(31, 666);
context.lineTo(31, 674);
context.stroke();
context.fillText("다온로드한", 55, 670);

function rightNumsArrow(){
function drawArrow(gap){
    context.strokeStyle = 'gray';
    context.beginPath();
    context.moveTo(380, 340 + gap);
    context.lineTo(387, 347 + gap);
    context.lineTo(380, 354 + gap);
    context.stroke();
}
}

```


위는 나머의 음악서랍 화면의 일부입니다 나머의 음악서랍 화면의 상단은 이미지를 로딩하는 함수를 만들어서 8 개의 이미지를 로딩하였는데 이는 메인화면에서 이미 설명하였기 때문에 생략하도록 하겠습니다.

의 부분은 아이콘과 아이콘의 설명을 구현한 화면과 코드입니다.

lineTo 와 arc 등을 이용하여 음표, 별, 하트, 문서아이콘, 등을 구현하였고, 옆에 아이콘들에 대한 설명을 적어놓았습니다.

하트 모양은 이전과 같이 두개의 반원과 두개의 직선을 이용하여,

내 플레이리스트 모양은 반복문을 사용하여 직선을 그리고, 원과 3 개의 직선을 이용해서 음표 모양을 그렸습니다

많이 들은 이라는 아이콘은 두개의 원을 반지름을 다르게 하여 구현하였습니다

팬맷은의 아이콘인 별은 가장 가장 그리기가 난해했던 모양이었습니다. 다른 도형들은 내각이 90 도등의 좌표 계산을 하기 쉬운 도형들이었지만 별은 각도가 다르기 때문에 좌표계산을 하는 것이 다른 도형에 비해 복잡하였습니다.

뮤직 DNA 는 큰 사각형을 그리고, 그 안에 반복문을 이용하여 x 축 좌표를 이동시키며, 직선의 길이도 증가시키면서 그렸습니다

다운로드 한 의 아이콘은 폴더 모양에 화살표가 있는 모양이라 좌표 설정에 약간 시간이 걸렸을 뿐 난이도가 높지는 않았습니



vangWoo

```
context.beginPath();
context.arc(320, 65, 6, Math.PI, 0, false);
context.lineTo(326, 72);
context.arc(320, 73, 6, 0, Math.PI, false);
context.lineTo(314, 65);
context.stroke();

context.beginPath();
context.moveTo(326, 69);
context.lineTo(314, 69);
context.stroke();

context.beginPath();
context.moveTo(320, 79);
context.lineTo(320, 83);
context.lineTo(314, 83);
context.lineTo(326, 83);
context.stroke();

//mic icon
context.beginPath();
context.arc(375, 70, 13, Math.PI * 3/2, Math.PI, false);
context.stroke();

context.beginPath();
context.arc(375, 70, 13, Math.PI + Math.PI * 1/18, Math.PI + Math.PI * 2/18, false);
context.stroke();

context.beginPath();
context.arc(375, 70, 13, Math.PI + Math.PI * 3/18, Math.PI + Math.PI * 4/18, false);
context.stroke();

context.beginPath();
context.arc(375, 70, 13, Math.PI + Math.PI * 5/18, Math.PI + Math.PI * 6/18, false);
context.stroke();

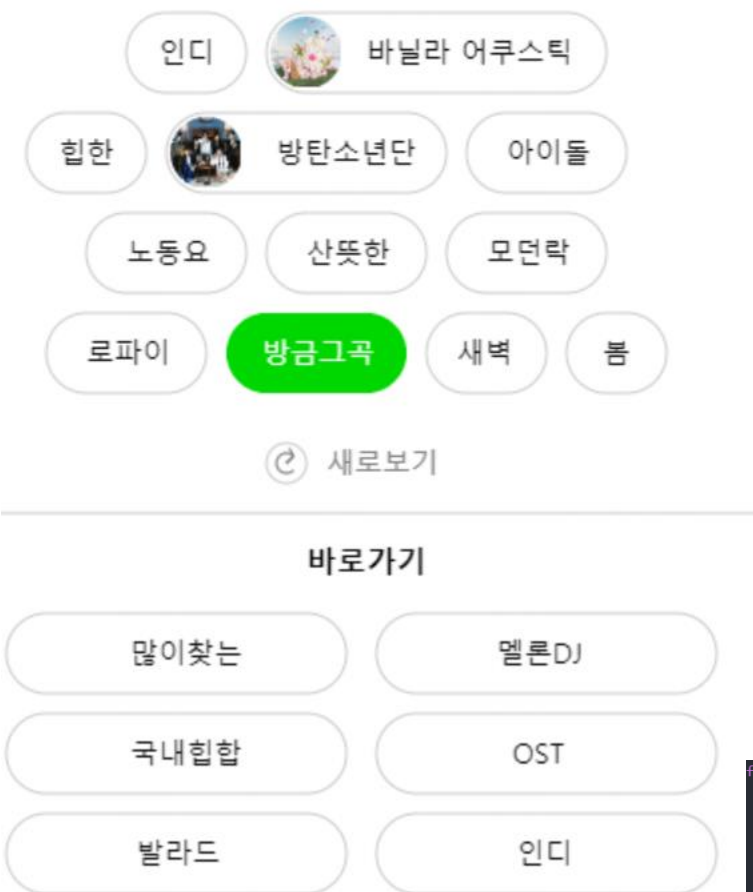
context.beginPath();
context.arc(375, 70, 13, Math.PI + Math.PI * 7/18, Math.PI + Math.PI * 8/18, false);
context.stroke();

context.beginPath();
context.arc(374, 72, 2, 0, Math.PI * 2, false);
context.fill();

context.beginPath();
context.moveTo(376, 72);
context.lineTo(376, 64);
context.lineTo(380, 67);
context.lineTo(380, 70);
context.stroke();
```

위 코드는 검색 화면에서의 우상단에 위치한 음악검색 아이콘과 마이크 아이콘을 만들기 위해 작성한 코드입니다. 마이크 아이콘은 여러 개의 직선과 두개의 반원을 이용하여 작성하였습니다

음악 검색 아이콘은 원과 3 개의 직선을 이용하여 내부의 음표를 완성하였고, 원을 3/4 만 그렸습니다. 그리고 나머지 1/4 부분에는 원 모양의 점들을 표시하기 위해 해당 부분의 각도를 약 10 등분을 하였고, 여러 번 원을 그려서 구현하였습니다.



```
function property(){
    let vanilla = new Image();
    vanilla.src = '../2017203020/AlbumCover/pretend.jpg';
    vanilla.onload=function(e){
        context.save();
        context.beginPath();
        context.arc(170,250,18, 0, Math.PI*2, false);
        context.closePath();
        context.clip();
        context.drawImage(vanilla, 152, 232,35,35);
        context.restore();
    }
    let bts=new Image();
    bts.src='../2017203020/AlbumCover/BTS.jpg';
    bts.onload=function(e){
        context.save();
        context.beginPath();
        context.arc(120,300,18, 0, Math.PI*2, false);
        context.closePath();
        context.clip();
        context.drawImage(bts, 102, 282,35,35);

        context.restore();
    }
}
```

```
function Border(xpos,ypos,length){
    let box={};
    box.xpos=xpos;
    box.ypos=ypos;
    box.length=length;
    box.draw=function(){
        context.beginPath();
        context.arc(this.xpos, this.ypos, 20, Math.PI/2, Math.PI * 3/ 2, false);
        context.lineTo(this.xpos+this.length, this.ypos-20);
        context.arc(this.xpos+this.length, this.ypos, 20, Math.PI * 3/ 2, Math.PI/2, false);
        context.lineTo(this.xpos, this.ypos+20);
        context.stroke();
    }
    return box;
}

function theme(){
    context.strokeStyle='rgba(204,204,204,1)';
    const b1= new Border(100, 250 ,20),
    b2= new Border(170,250,130),

    b3=new Border(50 , 300 ,20),
    b4=new Border(120 , 300 ,100),
    b5=new Border(270 , 300 ,40),

    b6=new Border(80 , 350 ,40),
    b7=new Border(170 , 350 ,40),
    b8=new Border(260, 350 ,40),

    b9=new Border(60, 400, 40),
    b10=new Border(250, 400, 20);
    b11=new Border(320, 400, 10)
    b1.draw();b2.draw();
    b3.draw();b4.draw();b5.draw();
    b6.draw();b7.draw();b8.draw();
}
```

위 부분은 나에게 맞춘 탐색 페이지의 주요 부분과 그 코드입니다. 우선 property 함수는 이전과 같이 이미지를 불러오는 함수입니다 하지만 이전과 다른 점은 이전의 이미지들은 이미지 소스 그대로 가져온다는 점이었는데 이 함수는 이미지를 가져오고, 그 이미지를 제가 원하는 모양(여기서는 원)으로 가공하여 이미지를 로딩한다는 점입니다. 이 함수에서는 총 두 번 이미지를 로딩하였습니다.

그리고 밑의 부분은 요구조건중에서 자바스크립트의 객체지향프로그래밍을 이용하여 객체, 생성자 함수, 변수, 함수 등을 정의하고 new 를 이용하여 객체들을 생성하는 모습입니다

해당 함수를 이용하여, 원래 어플에서는 버튼의 테두리 역할을 하였던 회색 도형을 입력받은 위치(x, y)와 크기를 이용하여 안에 글씨에 맞게 만들어 주었습니다.



```
function musicLists(){
    var musicOnPlayLists=[];
    function inst(src,title,singer,num){
        musicOnPlayLists[num] = new Image();
        musicOnPlayLists[num].src=src;
        musicOnPlayLists[num].onload=function(e){
            context.drawImage(musicOnPlayLists[num], 15, 160 + num * 58, 40,40);
        };
        context.fillStyle='white';
        context.font='15px malgun gothic';
        context.fillText(title, 65, 175 + num * 58);

        context.fillStyle='gray';
        context.font='14px malgun gothic';
        context.fillText(singer, 65, 197 + num * 58);
        i = 0;
        while(i < 3){
            context.beginPath();
            context.arc(370, 170 + i * 5 + num * 58, 1.5, 0, Math.PI * 2, false);
            context.fill();
            i++;
        }
    }
    inst('../2017203020/AlbumCover/healing_process.jpg', 'Good Night', '넬 (NELL)', 0);
    inst('../2017203020/AlbumCover/really_love.jpg', '정말로 사랑한다면', '버스커 버스커', 1);
    inst('../2017203020/AlbumCover/dynamite.jpg', 'Dynamite', '방탄소년단', 2);
    inst('../2017203020/AlbumCover/snowman.jpg', '눈사람', '정승환', 3);
    inst('../2017203020/AlbumCover/invitation.jpg', '폭풍속으로 (Feat. 버벌진트)', '에일리(AILEE)', 4);
    inst('../2017203020/AlbumCover/instagram.jpg', 'instagram', 'DEAN', 5);
    inst('../2017203020/AlbumCover/and_july.jpg', 'And July (Feat. DEAN)', '헤이즈 (Heize)', 6);
    inst('../2017203020/AlbumCover/roller_coaster.jpg', 'Roller Coaster', '청하', 7);
    inst('../2017203020/AlbumCover/pet.jpg', 'pet', '10cm', 8);
    inst('../2017203020/AlbumCover/boongboong.jpg', '뽕뽕 (Feat. 식케이)(Prod. GroovyRoom)', '김하온 (HAON)', 9);
}
```

위 화면은 마지막 화면인 플레이리스트 화면의 주요부분과 주요 코드입니다.

마지막 화면의 대부분은 이전에 제작하였던 도형들의 코드를 이용하여 여러 버튼들을 제작하였습니다

플레이리스트 화면의 주요 부분은 음악들이 있는 부분이라 할 수 있습니다 해당 부분은 우선 이전에 한 것과 같이 Image 객체를 생성하여 이미지를 받아왔고, 인자로 받은 노래 제목, 가수 이름을 출력하였습니다 그리고 맨 마지막에는 노래가 위치할 순서를 정하였고, 마지막으로 원래 어플에서는 터피하면 해당 음원의 자세한 정보를 보여주는 3 개의 회색 점까지 만드는 함수는 정의하였습니다. 이 부분 역시 원래는 생성자 함수 등을 통하여 정의하려고 하였으나, 그럴 경우 이미지가 로딩되지 않는 오류 때문에 다른 부분에서 정의하였습니다.

화면 별 객체 개수

메인화면 : 96, 재생중인음악 화면 : 52, 나만의 음악서랍 화면 : 84,

나에게 맞춘 탐색 화면 : 76 , 재생목록 화면 : 99

직선, 원, 사각형, 다각형(사각형이외), 배지어곡선,텍스트 비트맵 이미지는 메인 화면에서도 볼 수있듯이 직선 원, 사각형 텍스트, 비트맵 이미지는 하나 이상씩 사용하였고, 배지어 곡선은 재생중인 음악 화면에서 무작위재생 버튼을 구현하는데, 사각형 이외의 다각형은 재생, 다음 곡, 이전 곡 버튼의 삼각형, 그리고 나만의 음악서랍화면에서 별 을 구현하는데 사용하므로써, 요구조건을 만족시켰습니다

자바스크립트 객체지향 프로그래밍은 나에게 맞춘 탐색 부분에서 border 함수를 통하여 요구조건을 만족시켰습니다. 그리고 그 함수는 총 18 번 객체를 생성하였으며, 화면에서 회색 테두리 UI 를 담당하고 있습니다.

이벤트 핸들러는 부족하지만 화면 아무 부분이나 클릭시 다음 화면으로 넘어가도록 구현하였습니다.

외부 라이브러리 및 리소스는 사용하지 않았으며, 클라이언트 측 스크립트만 사용합니다. 또한 구글 크롬 웹에서 문제없이 구동되도록 하였습니다.

코드는 당연히 html 파일 1 개 안에 HTML, CSS, JavaScript 가 들어가 있습니다.



전반적인 자체 평가(성공적인 부분, 실패한 부분) 및 향후 개선점

저는 이 강의를 수강하면서 HTML/CSS/JavaScript 코딩을 처음 했습니다. 그래서 처음에 과제를 시작하면서 많이 막막하였으나, 강의자료와 구글링을 통하여 점점 Canvas 에 익숙해지면서 미숙하게나마 과제 1 을 완성할 수 있었습니다.

성공적인 부분 : 우선 단순하게 화면 모사는 나쁘지 않다고 생각합니다.

어플의 화면과 비슷한 느낌을 주려고 노력했고 깔끔하게 도형들을 만들려고 노력하였습니다.

실패한 부분 :

1. 버튼 : 저는 이벤트 핸들러를 이용하여 화면에서 버튼에 해당하는 화면을 누르면 그 화면으로 이동하게 구현하려고 시도하였습니다. 그

전에는 현재와 같이 화면의 아무 부분이나 누르면 지정한 화면으로 넘어가는 사이클이었습니다.

이벤트 핸들러를 구현하려고 시도한 결과 2 가지 큰 문제점이 발생하였습니다. 첫번째는 화면간 이동을 진행하다 보면 렉이 심하게 걸리면서 이미지 로딩이 점점 느려지다가 결국 화면이 멈추는 현상이 발생하였습니다. 두번째는 화면에서 호출하지 않은 이미지가 로딩되는 문제점이 발생하였습니다. 그 예로 재생중인음악 화면의 큰 앨범아트가 재생목록 화면에서도 로딩되고, 그 반대의 상황도 연출되었습니다.

2. 정적인 화면 : 제 과제는 클릭해서 다음 화면으로 넘어가는 이벤트핸들러 말고 다른 이벤트 핸들러가 존재하지 않습니다.

물론 함수로 화살표를 누르면 일시정지 아이콘으로 변경되게 만들려고 했으나, 역시 아이콘이 로딩되지 않아서 수정을 시도하였으나 역시 문제가 발생하여 수정을 포기하였습니다

향후 개선점 :

우선 실패한 부분의 버튼과 정적인 화면을 수정하고 싶습니다. 버튼을 누르는 것처럼 만들고 싶고, 실제 음악 플레이어와 조금이라도 더 유사하게 만들고 싶습니다. 또한 하위 resource 폴더에 음악을 하나라도 넣고 재생 버튼을 누르면 재생되게 하고 싶습니다.