Hello,

KDT 웹 개발자 양성 프로젝트

1기! 18th

with

지난 시간 리뷰

Sass

Import

Nesting

변수!($)

@mixin

@for

@each

@if

파노라마

UI 제작

https://panorama-ui-tetz.netlify.app/

Animation

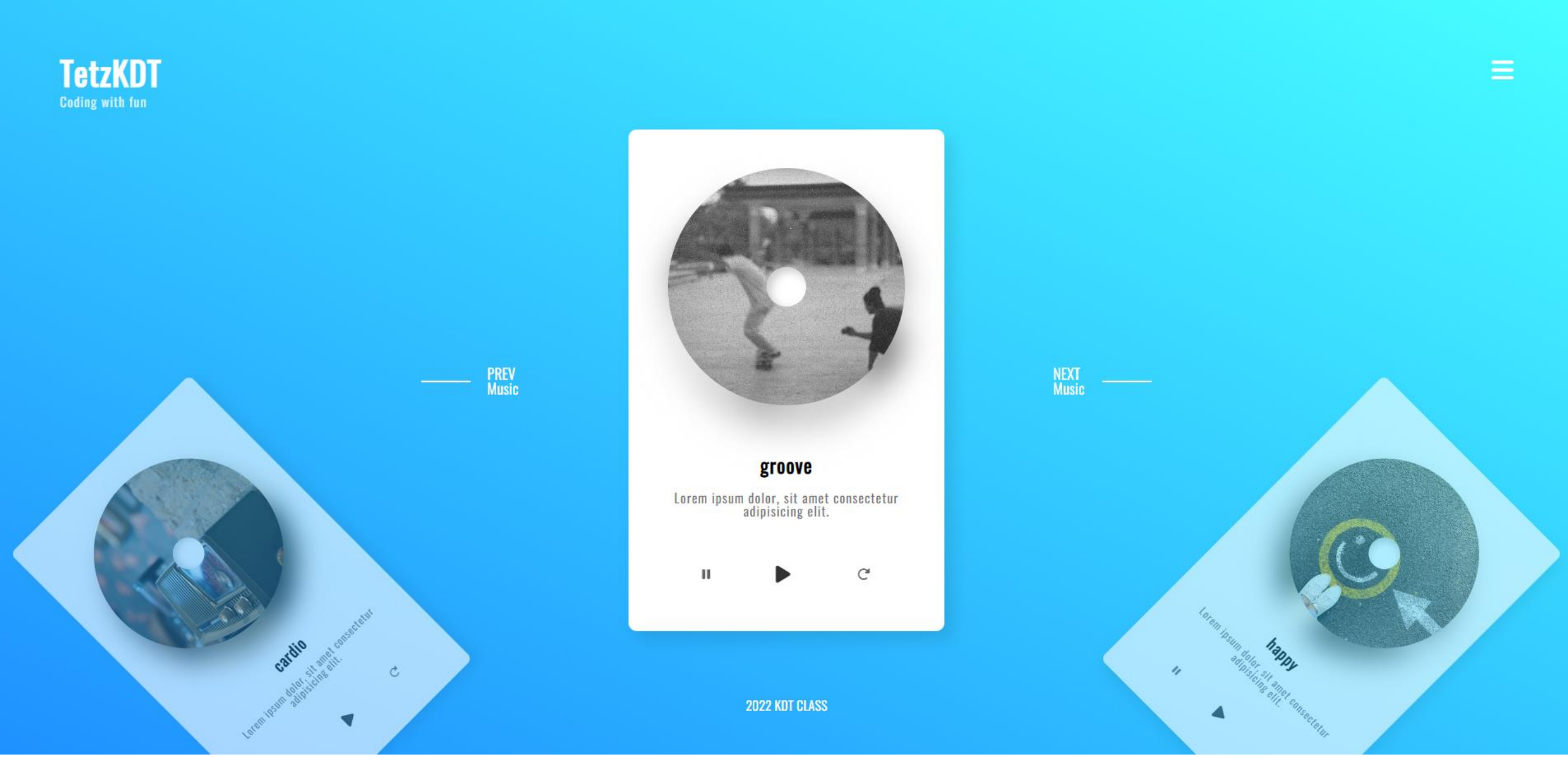
정지 기능 추가!

CSS 로 간단하게 처리해 봅시다!

```
&:hover {
      animation-play-state: paused;
    }
```

- #circle 에 hover 효과로 animation-play-state 속성 값을 paused 로 변경하기!

뮤직 플레이어 제작!



https://tetz-music-player.netlify.app/

기본 세팅하기~!

기본 세팅하기!

- 음악 및 이미지 파일 다운로드!
- 폰트 어썸 연결하기!
- JS 파일 연결하기!(Defer)
- Scss 파일 컴파일 → CSS 파일 링크!
- 폰트 가져오기
  - Oswald! (각자 원하는 것으로 가져오기!)
- _reset 설정

기본 UI

설정하기!

기본 UI, HTML 작업

- Figure 태그 하나 사용!
- H1 으로 로고 작업!
- A 태그로 bars 메뉴 설정(실제로 작동은 X)
- P 태그로 푸터 설정!

```
<figure>
        <h1>
            <strong>TetzKDT</strong><br>
            <span>Coding with fun</span>
        </h1>

        <a href="#" class="menu">
            <i class="fas fa-bars"></i>
        </a>

        <p>2022 KDT CLASS</p>
</figure>
```

기본 UI, CSS 작업!

- Body
  - 전체 폰트 설정
- Figure
  - 뷰포트 만큼 크기 주기
  - Overflow: hidden
  - Position: relative
  - Background: linear-gradient(25deg, dodgerblue, rgb(71, 255, 255));

```
body {
  font: 16px/1 "Oswald";
}

figure {
  width: 100%;
  height: 100vh;
  overflow: hidden;
  position: relative;
  background: linear-gradient(25deg, dodgerblue, rgb(71, 255, 255));
```

기본 UI, CSS 작업!

- H1
  - Position: absolute;
  - Top: 7vh / left: 4vw
  - 줄 간격을 없애기 위하여 → font-size: 0;
- Strong
  - Font: bold 40px/1.4 “Oswald”
  - Color: #fff / letter-spacing: 1px;
- Span
  - Font: 16px/1 “Oswald” / color: #fff / opacity: 0.8 / letter-spacing: 1px

```
h1 {
    position: absolute;
    top: 7vh;
    left: 4vw;
    font-size: 0;

    strong {
      font: bold 40px/1.4 "Oswald";
      color: #fff;
      letter-spacing: 1px;
    }

    span {
      font: 16px/1 "Oswald";
      color: #fff;
      opacity: 0.8;
      letter-spacing: 1px;
    }
  }
```

기본 UI, CSS 작업!

- .menu
  - Position: absolute / top: 8vh / right: 4vw
  - font-size: 30px / color: #fff
- > p {
  - Position: absolute
  - bottom: 7vh / left: 50% / tranfom: translate(-50%, 0);
  - color: #fff

```css
.menu {
  position: absolute;
  top: 8vh;
  right: 4vw;
  font-size: 30px;
  color: #fff;
}

> p {
  position: absolute;
  bottom: 7vh;
  left: 50%;
  transform: translate(-50%, 0);
  font: 16px/1 "Oswald";
  color: #fff;
}
```

음악 패널

작성하기!

패널, HTML 작업

- Section 태그 사용!
- 각각의 음악 패널은 article 로 작업
  - 각각 패널마다 transform 효과를 주어야 하므로 div.inner 를 배치하여 transform 이 중복되어 효과가 안걸리는 문제를 회피!

```
section>article*8>.inner
```

```
<section>
            <article>
                <div class="inner"></div>
            </article>
```

패널, CSS 작업

- 패널은 CSS 를 콤포넌트로 만들어 작업!
- _panel.scss 파일 작성
- Style.scss 파일에 panel 임포트

```
@import "panel";
```

패널, CSS 작업

- _panel.scss 파일에서 작업 시작!
- Section
  - Width: 20vw / height: 65vh
  - Position: absolute / left: 50% / top : 50% / margin-left: -10vw / margin-top: 32.5vh
- Article
  - Width: 100% / height: 100%
  - Position : absolute / top : 0px / left: 0px

```css
section {
  width: 20vw;
  height: 65vh;
  position: absolute;
  left: 50%;
  top: 150%;
  margin-left: -10vw;
  margin-top: -32.5vh;

  article {
    width: 100%;
    height: 100%;
    position: absolute;
    top: 0px;
    left: 0px;
```

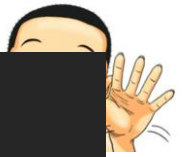
패널, CSS 작업

- .inner
  - Width: 100% / height: 100% / background-color: #fff
  - Padiing: 5vh 2.5vw 8vh / border-radius: 10px;
  - Box-shadow: 10px 10px 20px rgba(0, 0, 0, 0.1);
  - Opacity: 0.6

```css
.inner {
      width: 100%;
      height: 100%;
      background-color: #fff;
      padding: 5vh 2.5vw 8vh;
      border-radius: 10px;
      box-shadow: 10px 10px 20px rgba(0, 0, 0, 0.1);
      opacity: 0.6;
```

패널, 회전 주기

- 파노라마의 경우 nth-of-type() 를 이용해서 article 에 각각 회전 값을 주었었죠!?
- 이번에는 JS 를 통해서 각각의 회전 값을 한번에 처리해 봅시다!
- Section 요소 가져오기
- Section 내부의 article 요소 전부 가져오기
- Article 개수 선언 및 각도 구하기



```
const frame = document.querySelector("section");
const list = frame.querySelectorAll("article");
const len = list.length;
const deg = 360 / len;

for (let i = 0; i < len; i++) {
  list[i].style.transform = `rotate(${deg * i}deg`;
}
```

패널, 회전 주기

- 패널이 지금은 자기 자신의 중앙을 기준으로 회전 되어 있으므로 특정 기준 점을 기준으로 퍼져야 하므로 translateY 값 부여

```
for (let i = 0; i < len; i++) {
  list[i].style.transform = `rotate(${deg * i}deg) translateY(-100vh)`;
```

- 패널이 너무 위로 가있으므로!
- Section 의 top → 150%

```
section {
  width: 20vw;
  height: 65vh;
  position: absolute;
  left: 50%;
  top: 150%;
  margin-left: -10vw;
  margin-top: -32.5vh;
  transition: 1s;
```

좌우 이동

버튼 만들기!

버튼, HTML 작업

- Section 태그 아래에 nav 태그로 작업!
- 이전, 이후 버튼이 있어야 하므로! 2개 작업

```
<nav class="btnPrev">
     <span>PREV Music</span>
</nav>

<nav class="btnNext">
    <span>NEXT Music</span>
</nav>
```

버튼, CSS 작업

- 버튼도 콤포넌트로 작업!
- _btns.scss 파일 생성
- Style.scss 파일에 임포트!

```
@import "panel";
@import "btns";
```

버튼, CSS 작업

- .btnPrev
  - Width: 60px / height: 60px
  - Position: absolute / top: 50% / left: 50%
  - Transform(-20vw, -50%)
  - Display: flex / justify-content: felx-start / align-itmes: center
  - Font-size: 0 / padding-left:20px / cursor: pointer
- Span
  - Font-size: 18px / color: #fff / transition: 0.5s

```css
.btnPrev {
  width: 60px;
  height: 60px;
  position: absolute;
  top: 50%;
  left: 50%;
  transform: translate(-20vw, -50%);
  display: flex;
  align-items: center;
  justify-content: flex-start;
  cursor: pointer;
  font-size: 0;
  padding-left: 20px;

  span {
    font-size: 18px;
    color: #fff;
    transition: 0.5s;
  }
```

버튼, 움직이는 화살표 만들기

- 요런거는 보통 가상 요소 선택자로 만듭니다!
- ::Before / ::after 둘 다 선언해서 만들고 Ani 효과 주기
- ::Before
  - Content: “”
  - Display: block / width: 100% / height: 2px / background-color: #fff
  - Position: absolute / left: 0px
  - Transform-origin: left center / transform: rotate(-180deg)
  - Transition: 0.5s

```scss
&::before,
&::after {
    content: "";
    display: block;
    width: 100%;
    height: 2px;
    background-color: #fff;
    position: absolute;
    left: 0px;
    transform-origin: left center;
    transform: rotate(-180deg);
    transition: 0.5s;
  }
```

```scss
&::after {
    transform: rotate(180deg);
}
```

버튼, 움직이는 화살표 만들기

- Hover 효과 주기
  - 글자는 쇽 사라지게!
  - ::Before / ::After 는 각도를 변경해서 화살표처럼 보이도록
    - 30도를 기준으로 위 아래로 위치 하도록 설정!

```
&:hover {
    span {
      transform: translate(100%, 0);
      opacity: 0;
    }

    &::before {
      transform: rotate(-30deg);
    }

    &::after {
      transform: rotate(30deg);
    }
  }
```

실습, Next 버튼도 작업해 봅시다!

- Prev 버튼과 거의 유사한 Next 버튼 작업하기!

좌우 이동 버튼

클릭 액션 처리

버튼, JS 처리!

- 버튼을 각각 불러옵시다!
- 해당 패널의 위치 값을 저장할 변수 만들기!
- 각각 버튼에 이벤트 붙이기!
- 0을 기준으로 next 를 누르면 +1 * 45deg 씩 돌아가고
- Prev 를 누르면 -1 * 45deg 씩 돌아가면 됩니다!
- 다만, 해당 속성 값은 누적이 아니므로 next 가 2번 눌리면 45deg 에서 90deg 로 돌아야 합니다!
- 즉, 해당 패널의 위치 값을 0 을 기준으로 정해서 곱하면 됩니다!

```
const prev = document.querySelector(".btnPrev");
const next = document.querySelector(".btnNext");
let num = 0;

prev.addEventListener("click", function (e) {
  frame.style.transform = `rotate(${deg * ++num}deg)`;
});

next.addEventListener("click", function (e) {
  frame.style.transform = `rotate(${deg * --num}deg)`;
});
```

- 돌아가는 애니메이션을 줘야 하는데, 돌아가는 UI는 Section 이므로 Section 에 transition: 1s; 값 추가

가운데 패널에 활성화 주기

활성화 패널, on 클래스 부여

- 첫번째 패널인 Article 에 on 클래스를 부여

```
<section>
            <article class="on">
                <div class="inner">
```

- On 클래스를 가진 패널은
  - Opacity: 1 / transform: scale(1) 부여

```
&.on {
        .inner {
          opacity: 1;
          transform: scale(1);
        }
    }
```

활성화 패널, JS 처리하기

- 전역 변수로 활성화 패널의 위치를 저장 → let active = 0;
- Prev 버튼 처리
  - 초창기 위치가 0
  - 0에서 prev 가 눌리면? → 제일 마지막(article 의 개수 – 1)으로 이동
  - 그 외에는 자신의 값에서 -1 만 해주면 됩니다!
  - 그리고 현재 모든 article 에서 on 이라는 클래스를 제거 후 → active 위치에 만 on 클래스 추가!

```
let active = 0;

prev.addEventListener("click", function (e) {
  frame.style.transform = `rotate(${deg * ++num}deg)`;

  if (active === 0) {
    active = len - 1;
  } else {
    active--;
  }

  for (let el of list) {
    el.classList.remove("on");
  }
  list[active].classList.add("on");
});
```

활성화 패널, JS 처리하기

- Next 버튼 처리
  - 이 친구는 반대로, 마지막 위치에서 next 가 눌리면 0으로 변경!
  - 나머지는 클릭 되면 +1 해주면 되겠죠?
  - 역시 모든 article 에서 on 클래스 제거 후 → active 위치에 on 클래스 부여

```
next.addEventListener("click", function (e) {
  frame.style.transform = `rotate(${deg * --num}deg)`;

  if (active === len - 1) {
    active = 0;
  } else {
    active++;
  }

  for (let el of list) {
    el.classList.remove("on");
  }
  list[active].classList.add("on");
});
```

패널

꾸미기

앨범 사진, HTML 작업

- 먼저 앨범 사진을 넣어 보아요!
- 사진을 넣을 div 요소 pic 삽입!
- CD 처럼 보이기 위한 가운데 구멍 dot 도 삽입!

```
<section>
            <article class="on">
                <div class="inner">
                    <div class="pic">
                        <div class="dot"></div>
                    </div>
                </div>
             </article>
```

앨범 사진, CSS 작업

- .pic
  - Width: 15vw / height: 15vw;
- 실제 사진은 ::before 와 ::after 를 사용해서 처리 합니다!
  - Content: “”
  - Display: block / width: 100% / height: 100%
  - position: absolute / top: 0px / left: 0px / border-radius: 50%
  - background-repeat: no-repeat / background-position: center / background-size: cover
  - 이미지 주소로 이미지 삽입!

```
.pic {
        height: 15vw;
        width: 15vw;
        position: relative;

        &::before,
        &::after {
          content: "";
          display: block;
          width: 100%;
          height: 100%;
          position: absolute;
          top: 0px;
          left: 0px;
          border-radius: 50%;
          background-repeat: no-repeat;
          background-position: center;
          background-size: cover;
          background-image: “url()”;
        }
```

앨범 사진, CSS 작업

- 지금은 사진 2장이 겹쳐진 상태이죠?
- ::before 만 따로 이미지를 처리해서 자연스러운 그림자 효과를 줍시다
  - Transform: translaste(0, 10%) / filter: blur(20px) brightness(130%);
- .dot 가운데 구멍도 처리해 보죠
  - Width: 2.5vw / height: 2.5vw / postion: absolute / top: 50% / left: 50% / transform: translate(-50%, -50%) / background-color: #fff / border-radius: 50% / box-shadow: inset 5px 5px 10px rgba(0, 0, 0, 0.2); / z-index: 3

```
::before {
          transform: translate(0, 10%);
          filter: blur(20px) brightness(130%);
        }

        .dot {
          width: 2.5vw;
          height: 2.5vw;
          position: absolute;
          top: 50%;
          left: 50%;
          transform: translate(-50%, -50%);
          background-color: #fff;
          border-radius: 50%;
          box-shadow: inset 5px 5px 10px rgba(0, 0, 0, 0.2);
          z-index: 3;
        }
```

앨범 사진, JS로 일괄 삽입을 해봅시다!

- 현재 앨범 사진과 음악 mp3 파일명은 동일한 상태이죠?
- JS에서 배열을 사용해서 일괄적으로 이미지를 삽입해 Boa요!
  - 각각의 파일명이 담긴 배열 만들기
  - 이미 사용한 반복문이 있으니, 해당 반복문에 처리를 해봅시다!

```
const names = ["cardio", "groove", "happy", "light", "lily", "limes", "pop", "swing"];

for (let i = 0; i < len; i++) {
  list[i].style.transform = `rotate(${deg * i}deg) translateY(-100vh)`;

  const pic = list[i].querySelector(".pic");
  pic.style.backgroundImage = `url(../img/${names[i]}.jpg)`;
}
```

앨범 사진, before / after 에 상속 시키기

- 지금은 .pic 에 사진이 들어간 상태입니다!
- .pic 사진을 저 멀리 먼저 치웁시다
  - Background-repeat: no-repeat / background-position: -9999px -9999px;
- 그리고 ::before ::after 에 .pic 의 사진 위치를 상속 시킵니다!

```scss
.pic {
        height: 15vw;
        width: 15vw;
        position: relative;
        background-repeat: no-repeat;
        background-position: -9999px -9999px;

        &::before,
        &::after {
          content: "";
          display: block;
          width: 100%;
          height: 100%;
          position: absolute;
          top: 0px;
          left: 0px;
          border-radius: 50%;
          background-repeat: no-repeat;
          background-position: center;
          background-size: cover;
          background-image: inherit;
        }
```

앨범 제목, HTML 영역

- .text 영역 만들기
  - H2
  - P

```
<section>
            <article class="on">
                <div class="inner">
                    <div class="pic">
                        <div class="dot"></div>
                    </div>
                    <div class="text">
                        <h2></h2>
                        <p>Lorem ipsum dolor, sit amet consectetur.</p>
                    </div>
                </div>
            </article>
```

앨범 제목, JS로 제목도 일괄 적용

- 이미 선언한 이름 배열을 또 쓰면 되겠죠?

```
const names = ["cardio", "groove", "happy", "light", "lily", "limes", "pop", "swing"];

for (let i = 0; i < len; i++) {
  list[i].style.transform = `rotate(${deg * i}deg) translateY(-100vh)`;

  const pic = list[i].querySelector(".pic");
  pic.style.backgroundImage = `url(../img/${names[i]}.jpg)`;

  const title = list[i].querySelector(".text>h2");
  title.innerHTML = `${names[i]}`;
}
```

앨범 제목, text 영역 CSS 작업

- .text
  - Width: 15vw / text-align: center
  - Position: absolute / margin-top: 60px / letter-spacing: 1px;
- H2
  - Margin-bottom: 20px;
- P
  - Color: #777;

```
.text {
        width: 15vw;
        text-align: center;
        position: absolute;
        margin-top: 60px;
        letter-spacing: 1px;

        h2 {
          margin-bottom: 20px;
        }

        p {
          color: #777;
        }
```

뮤직 플레이어, HTML 작업

- UI 태그 + icon 으로 뮤직 플레이어를 만듭시다!
- UI > li * 3
  - 각각 li 에 icon 으로 버튼 주기

```
<div class="text">
    <h2></h2>
    <p>Lorem ipsum dolor, sit amet consectetur adipisicing elit.</p>
    <ul class="control">
        <li class="pause"><i class="fas fa-pause"></i></li>
        <li class="play"><i class="fas fa-play"></i></li>
        <li class="reload"><i class="fas fa-redo-alt"></i></li>
    </ul>
</div>
```

뮤직 플레이어, CSS 작업

- UI
  - 가로 배치죠? → display: flex
  - 버튼이 적절히 퍼지도록 → justify-content: space-around
- UI li
  - Cursor: pointer;
  - Opacity: 0.6 / transition: 0.5s
- &.play
  - 플레이 버튼은 더 크고 선명하게 보이도록 → transform: scale(1.5) / opacity: 0.9;

```scss
.control {
          display: flex;
          justify-content: space-around;
          margin-top: 60px;

          li {
            cursor: pointer;
            opacity: 0.6;
            transition: 0.5s;

            &.play {
              transform: scale(1.5);
              opacity: 0.8;
            }
```

뮤직 플레이어, CSS 작업

- Hover 효과 주기
  - 각각의 버튼 → 1.5배 / opacity: 0.8
  - Play 버튼 → 1.8배 / opacity: 1

```
&:hover {
              transform: scale(1.5);
              opacity: 0.8;

              &.play {
                transform: scale(1.8);
                opacity: 1;
              }
            }
```

앨범 이미지 회전

앨범 회전, Keyframes 작업

- 앨범은 ::before 와 ::after 2개의 이미지가 있으므로 2개의 keyframes 작업 필요
- 하나는 자신의 위치에서 360deg 돌고
- 하나는 자신의 위치 10% 아래에서 360deg 돌면 되므로!

```
@keyframes rotation {
  0% {
    transform: rotate(0deg);
  }

  100% {
    transform: rotate(360deg);
  }
}

@keyframes rotation2 {
  0% {
    transform: translateY(10%) rotate(0deg);
  }

  100% {
    transform: translateY(10%) rotate(360deg);
  }
}
```

앨범 회전, CSS 작업

- 사진 요소에 on 이라는 클래스가 부여되면 회전 하도록 설정!

```
.pic {
    &.on {
          &::before {
            animation: rotation2 4s linear infinite;
          }

          &::after {
            animation: rotation 4s linear infinite;
          }
        }
```

앨범 회전, JS 작업

- Play 버튼을 누르면 on 클래스가 부여 되도록 처리!
- Pause 버튼을 누르면 on 클래스가 제거 되도록 처리!

```
for (let el of list) {
  const play = el.querySelector(".play");
  const pause = el.querySelector(".pause");
  const load = el.querySelector(".reload");

  play.addEventListener("click", function (e) {
    e.currentTarget.closest("article").querySelector(".pic").classList.add("on");
  });

  pause.addEventListener("click", function (e) {
    e.currentTarget.closest("article").querySelector(".pic").classList.remove("on");
});
```

음악 파일

삽입 및 재생

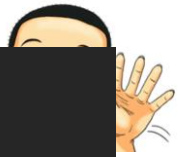
음악 파일 , JS로 일괄 삽입을 해봅시다!

- 아까 쓰던 반복문이 있네요!?
- createElement 로 audio 엘리먼트 만들기
- setAttribute로 위치(src) 속성 입력하기
- 재생이 무한히 반복해야 하므로 loop 속성도 주기
- 각각의 list 요소에 audio 붙여넣기!

```
for (let i = 0; i < len; i++) {
  list[i].style.transform = `rotate(${deg * i}deg) translateY(-100vh)`;

  const pic = list[i].querySelector(".pic");
  pic.style.backgroundImage = `url(../img/${names[i]}.jpg)`;

  const title = list[i].querySelector(".text>h2");
  title.innerHTML = `${names[i]}`;

  const audio = document.createElement("audio");
  audio.setAttribute("src", `../music/${names[i]}.mp3`);
  audio.setAttribute("loop", "loop");

  list[i].append(audio);
}
```

음악 파일, JS로 재생 처리!

- 아까 play 랑 pause 버튼 작업 했었죠!? 거기로 고고싱
- 재생 버튼을 클릭하면 audio 태그를 선택 후 play() 메소드로 재생
- 멈춤 버튼을 클릭하면 audio 태그를 선택 후 pause() 메소드로 멈춤
- 이번에는 reload 버튼도 작업 합시다!
  - 재생과 완전히 동일하지만 처음부터 틀어야 하므로
  - Load() 로 음악을 다시 로드 시킨 후 → play()



```
for (let el of list) {
  const play = el.querySelector(".play");
  const pause = el.querySelector(".pause");
  const load = el.querySelector(".reload");

  play.addEventListener("click", function (e) {
    e.currentTarget.closest("article").querySelector(".pic").classList.add("on");
    e.currentTarget.closest("article").querySelector("audio").play();
  });

  pause.addEventListener("click", function (e) {
    e.currentTarget.closest("article").querySelector(".pic").classList.remove("on");
    e.currentTarget.closest("article").querySelector("audio").pause();
  });

  load.addEventListener("click", function (e) {
    e.currentTarget.closest("article").querySelector(".pic").classList.add("on");

    e.currentTarget.closest("article").querySelector("audio").load();
    e.currentTarget.closest("article").querySelector("audio").play();
  });
}
```

음악 파일, 여러 개가 한꺼번에 재생?

- 다른 노래가 재생 되면 다른 노래가 멈추도록 작업해 봅시다!!
- 이전 노래의 위치를 저장할 before 라는 전역 변수 생성!
- Before 의 초기 값을 0으로 세팅하고 before 가 0일 경우 현재 패널의 위치를 저장
- 다른 패널에서 재생 또는 Reload 버튼을 눌렀을 경우 이전 위치의 노래를 정지
  - 현재 클릭이 된 패널의 위치와 before 를 비교하여 서로 다를 경우 before 의 노래를 정지 → 현 위치를 before 로 등록!

```
play.addEventListener("click", function (e) {
    if (before === 0) {
      before = e.currentTarget;
    } else if (before !== e.currentTarget) {
      before.closest("article").querySelector("audio").pause();
      before.closest("article").querySelector(".pic").classList.remove("on");
      before = e.currentTarget;
    }

    e.currentTarget.closest("article").querySelector(".pic").classList.add("on");

    e.currentTarget.closest("article").querySelector("audio").play();
  });
```

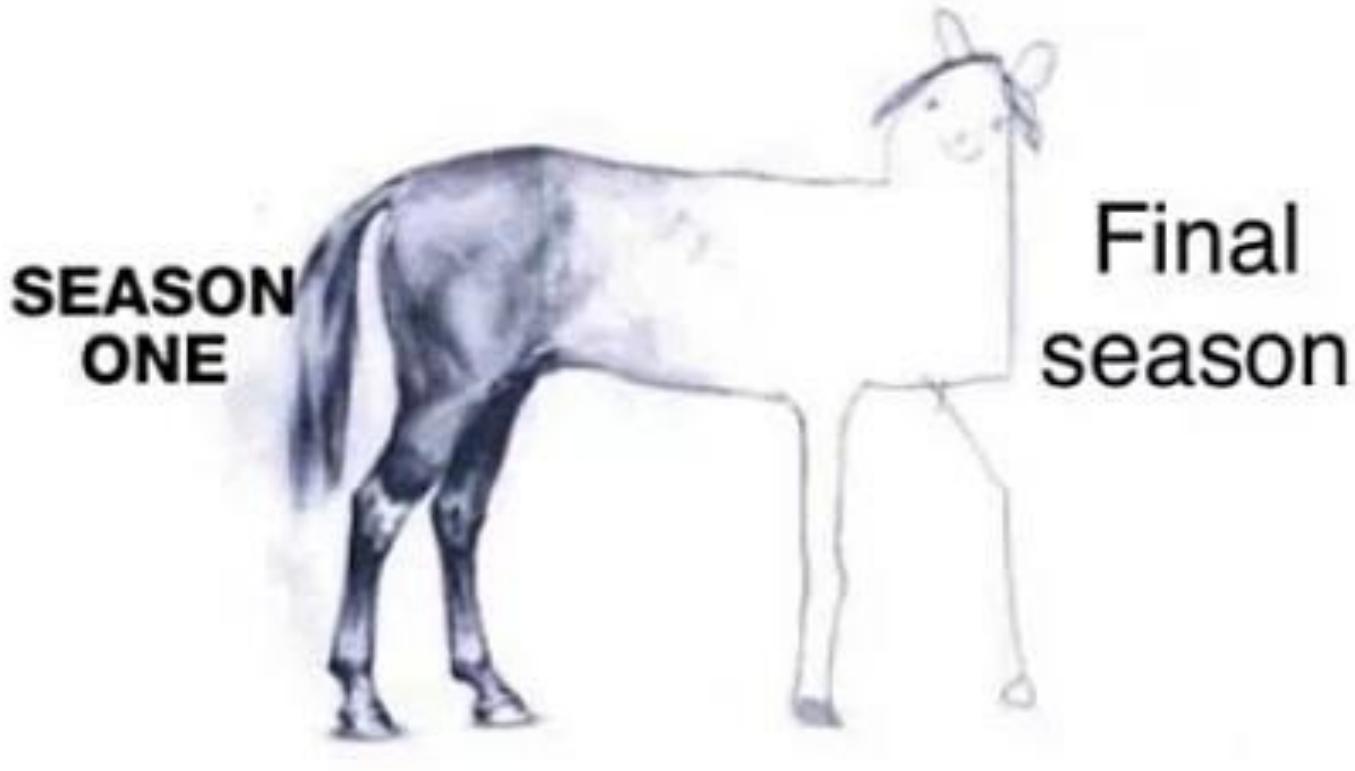

WE DID IT!
WOOHOO

2차

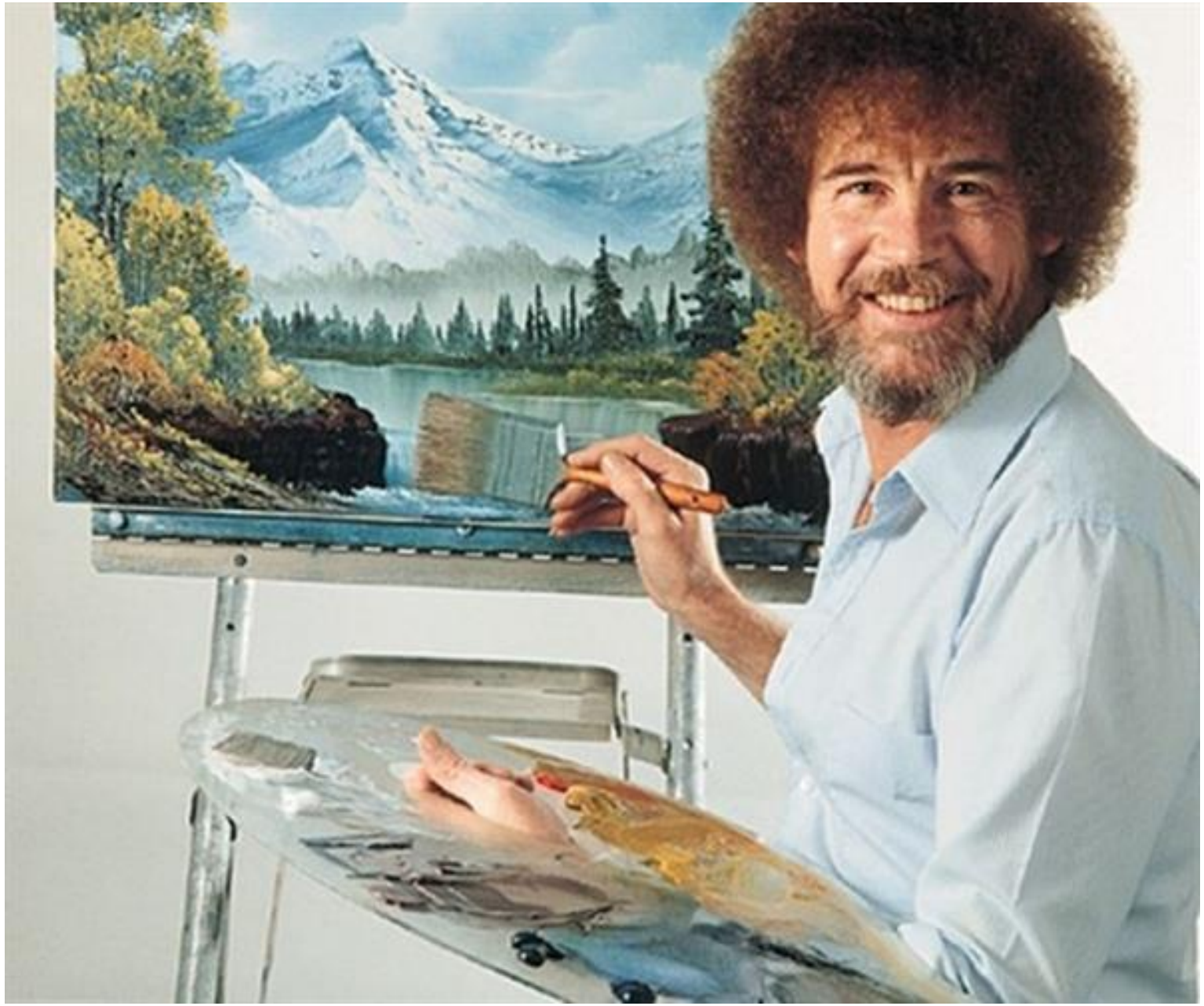
팀 프로젝트!!


SATURDAY NIGHT LIVE
tvN
HD
19
그럼 조장부터 정할까요?
조별 과제 잔혹사 2

GAME OF THRONES




3
5
JS



과제, 프로젝트 기획안 작성!

- 최대한 간단하게 작성해 주세요!
- 참고 페이지 등을 링크하시면 더 좋습니다!
- 1인당 최소 1개의 페이지를 작업 후, 합치는 방향으로 준비해 주세요.
- 기능은 최대 공동 데이터 API 가져오기 정도까지 하는게 좋을 것 같습니다!
- 해당 내용은, 수요일 수업으로 준비 할게요!


수고하셨습니다!