

**three.js**

---

01 three.js?

02 설치 방법 및 간단한 구현

03 기업에서 요구하는 이유

04 마무리

## 01 three.js?

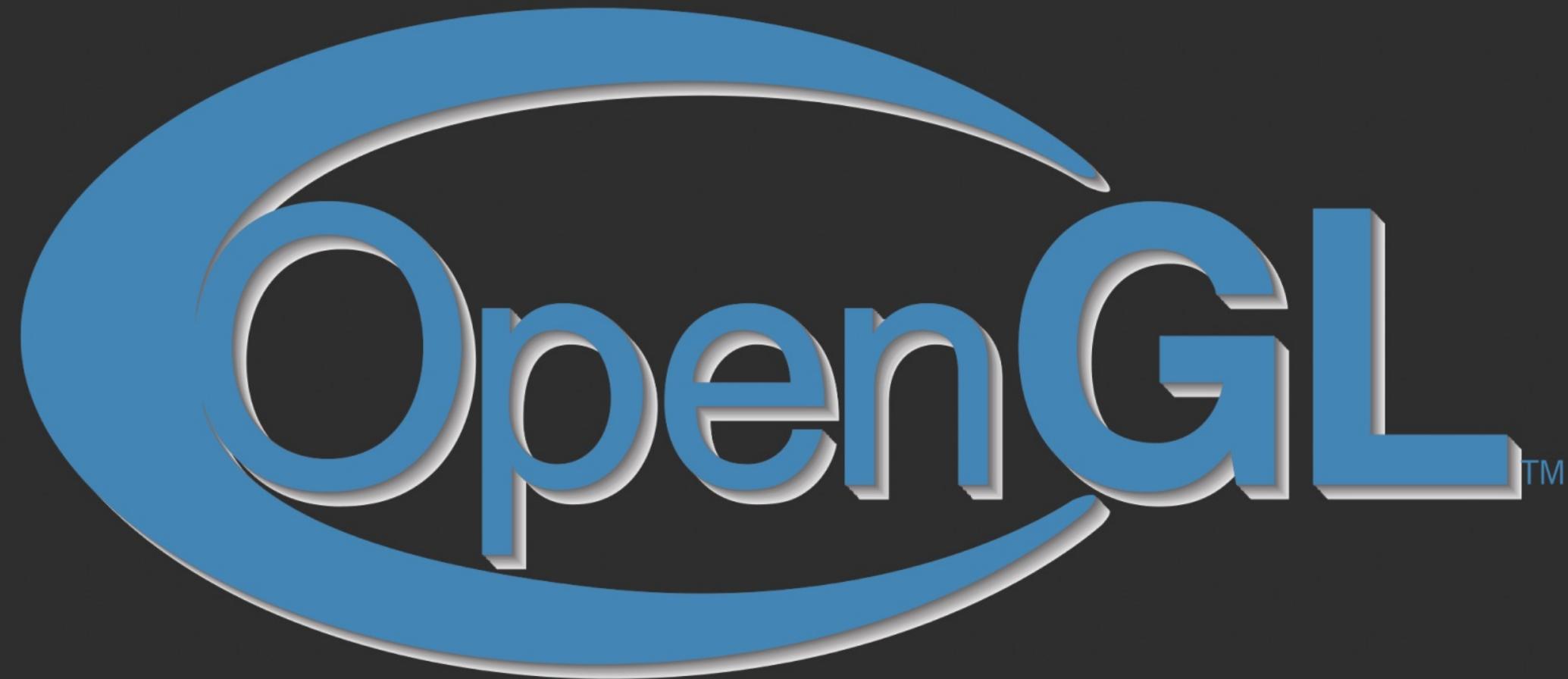
---



옛부터 웹에서 3D객체를 렌더링하려는 시도는 많았으나,  
호환성, 보안성, 편리성 등의 문제가 있어서 사용이 감소한 상태

## 01 three.js?

---



OpenGL은 그래픽 및 이미지를 조작하는 데 사용할 수 있는 많은 기능들을 제공하는 API  
C 기반으로 작성

## 01 three.js?

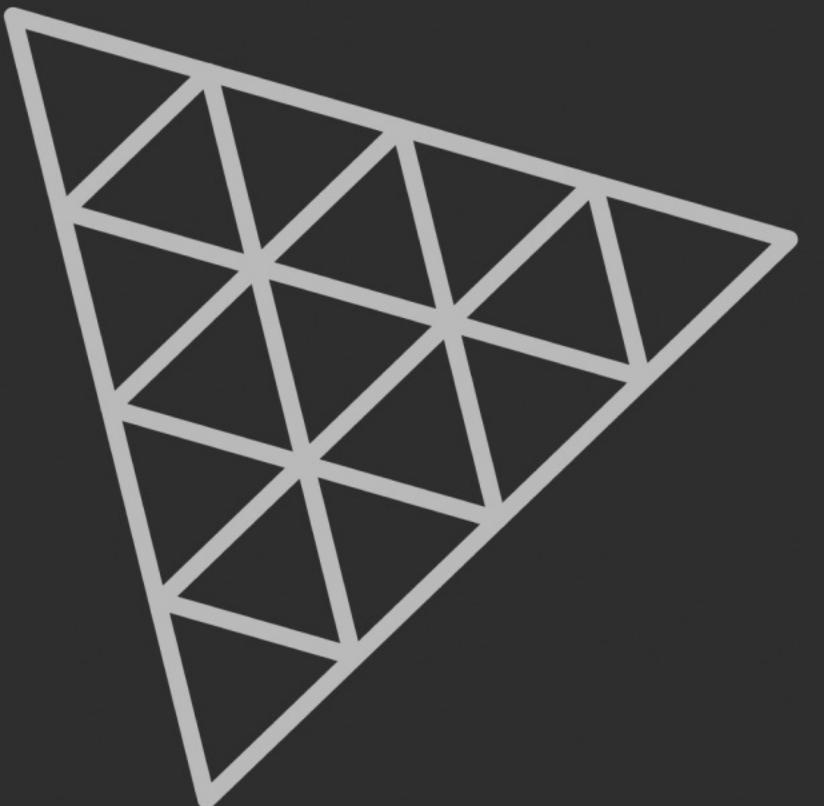
---



WebGL은 플러그인을 사용하지 않고 OpenGL ES 2.0 기반 API를 이용하여  
브라우저의 HTML canvas에 렌더링하여 3D 웹 콘텐츠 제작을 가능하게하는  
JavaScript 라이브러리

## 01 three.js?

---

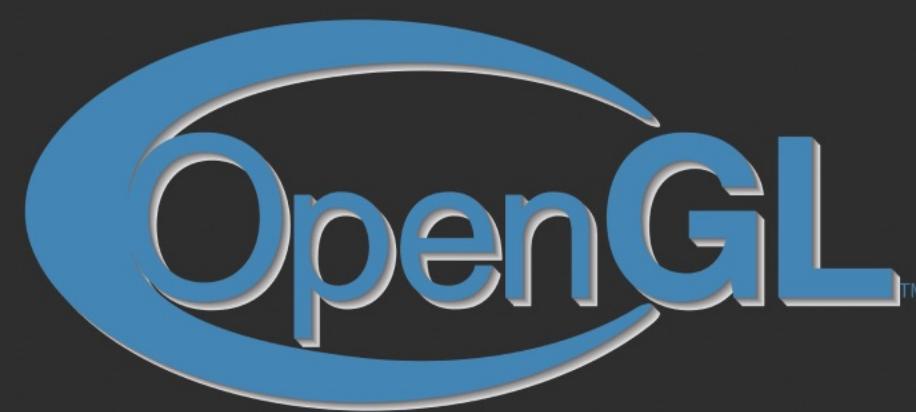


three.js

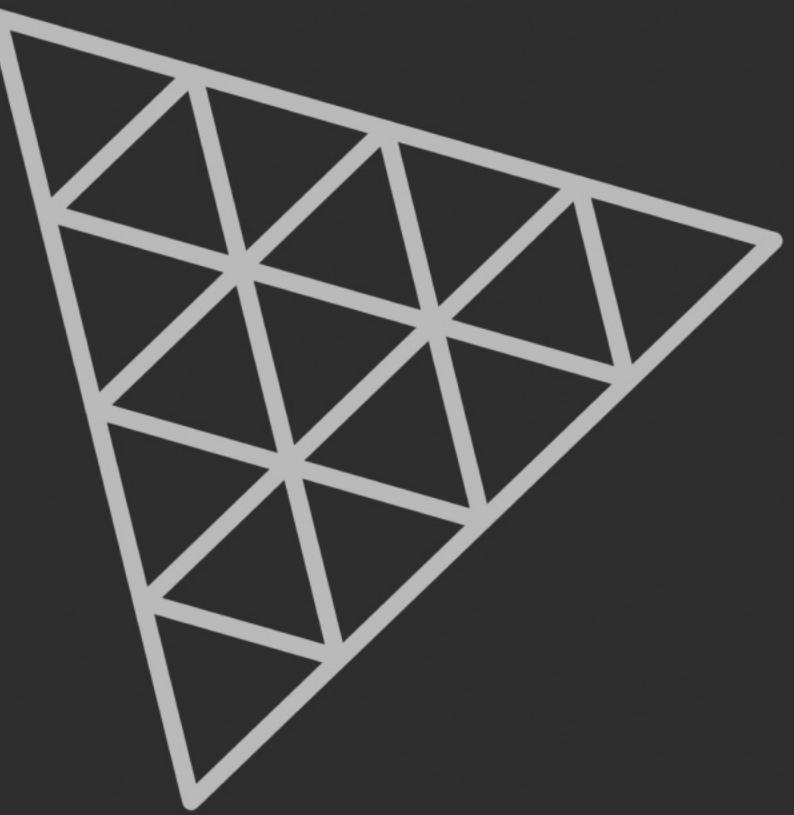
웹페이지에 3D 객체를 쉽게 렌더링해주는 라이브러리  
점, 선, 삼각형을 그리는 시스템인 WebGL을 이용한다

## 01 three.js?

---



three.js

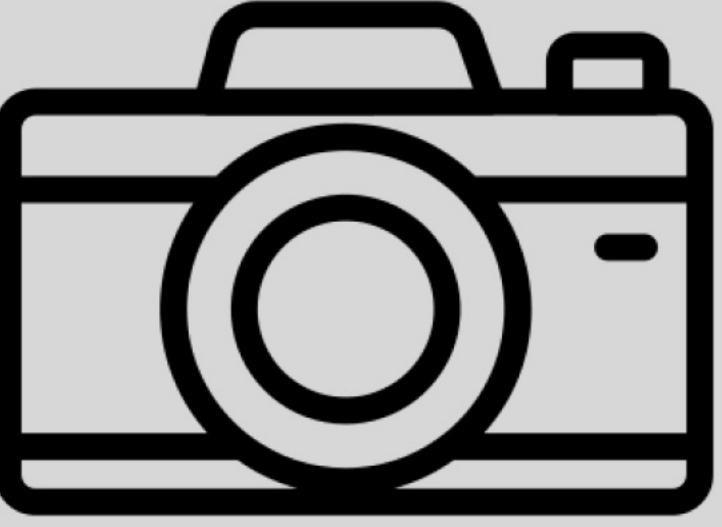


# 01 three.js?

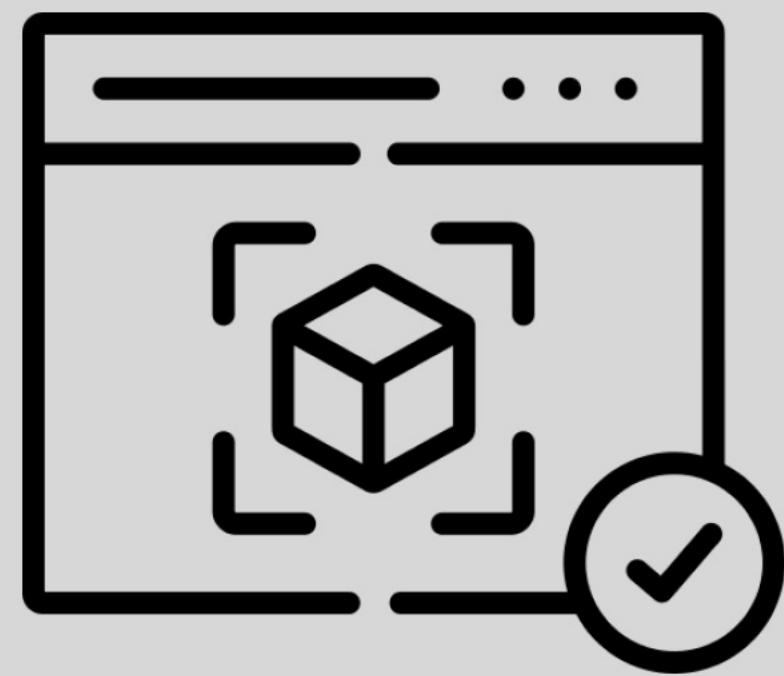
---



Scene



Camera



Renderer



Light

## 02 설치 방법 및 간단한 구현

---

### 1. 빌드 툴을 통한 설치

```
# three.js  
npm install --save three  
  
# vite  
npm install --save-dev vite
```

### 2. CDN을 통한 설치

```
<script type="importmap">  
{  
  "imports": {  
    "three":  
      "https://cdn.jsdelivr.net/npm/three@<version>/build/three.module.js",  
    "three/addons/":  
      "https://cdn.jsdelivr.net/npm/three@<version>/examples/jsm/"  
  }  
</script>
```

### 3. 모듈 불러오기

```
import * as THREE from 'three';  
import { OrbitControls } from 'three/addons/controls/OrbitControls.js';  
import { GLTFLoader } from 'three/addons/loaders/GLTFLoader.js';
```

### • 장면과 카메라 설정

```
const scene = new THREE.Scene();
const camera = new THREE.PerspectiveCamera(50, window.innerWidth /
window.innerHeight, 0.1, 10000);
camera.position.z = 5;
```

SCENE은 이미지 객체를 담은 컨테이너,  
도화지의 일종

### 가장 많이 사용되는 PerspectiveCamera

주요 매개변수는 아래와 같다

1. **fov** (Field of View, 시야각): 카메라로 보는 시야의 범위를 각도로 정의 (45~75가 자연스러움)
2. **aspect**(종횡비): 렌더링 장면의 가로세로 비율을 정의 (window는 브라우저 창 크기에 맞게 정의한 것)
3. **near** (근거리 클리핑 평면): 얼마나 가까운 물체부터 장면을 그릴지 정의 (너무 작으면 문제있음)
4. **far** (원거리 클리핑 평면): 얼마나 먼 물체까지 그릴지 정의 (1000 또는 그 이상의 값 사용)

### • 렌더러 설정

```
// 렌더러 생성 및 배경색 설정
const renderer = new THREE.WebGLRenderer({ antialias: true });
renderer.setClearColor(0x6A5ACD); // 배경을 보라색으로 설정
renderer.setSize(window.innerWidth, window.innerHeight);
```

3차원 객체를 2D에서 3D처럼 보이도록  
시각화 해주는 역할

1. **antialias** : 가장자리를 부드럽게, 계단 현상을 방지
2. **setClearColor** : 이미지 배경 색 선택, Background Color 역할
3. **setSize** : 이미지가 렌더링될 크기를 설정

- GLTF 모델 로드

```
// GLTF 모델 로드
const loader = new GLTFLoader();
loader.load('/scene.gltf', (gltf) => {
  scene.add(gltf.scene);
  animate();
});
```

### GLTF 파일 형식

3D 모델과 애니메이션을 저장하기 위한 표준 포맷으로,  
웹에서 효율적으로 처리할 수 있도록 설계  
JSON 형식으로 저장

### bin 파일 형식

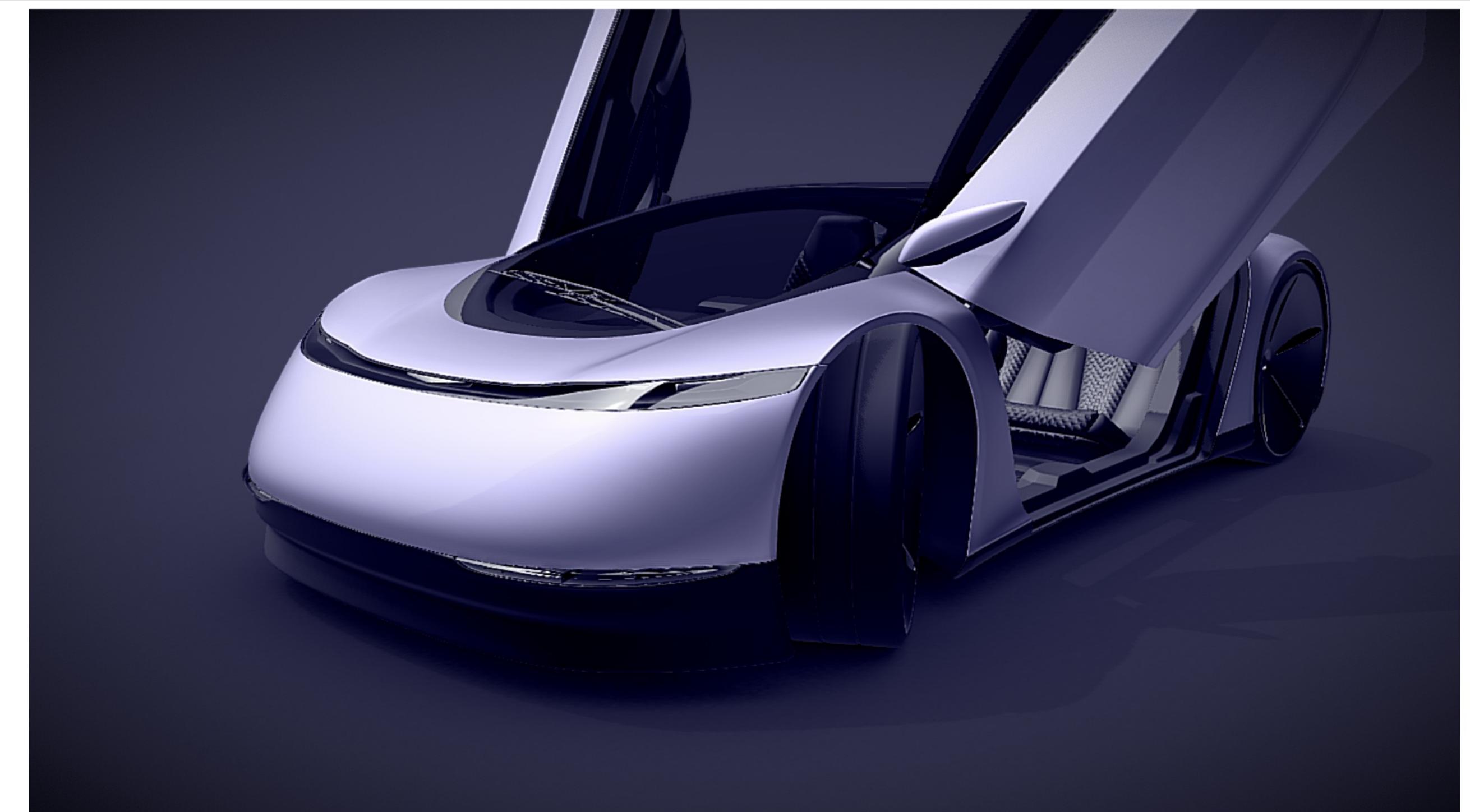
gltf 파일과 함께 사용되며, 모델의 정점 데이터(Vertex Data)  
나 텍스처 같은 바이너리 데이터를 담고 있음  
gltf는 이 데이터를 참조하여 정보를 로드함

### textures 폴더

모델에 입힐 텍스처 이미지가 들어있는 폴더

## 02 설치 방법 및 간단한 구현

---

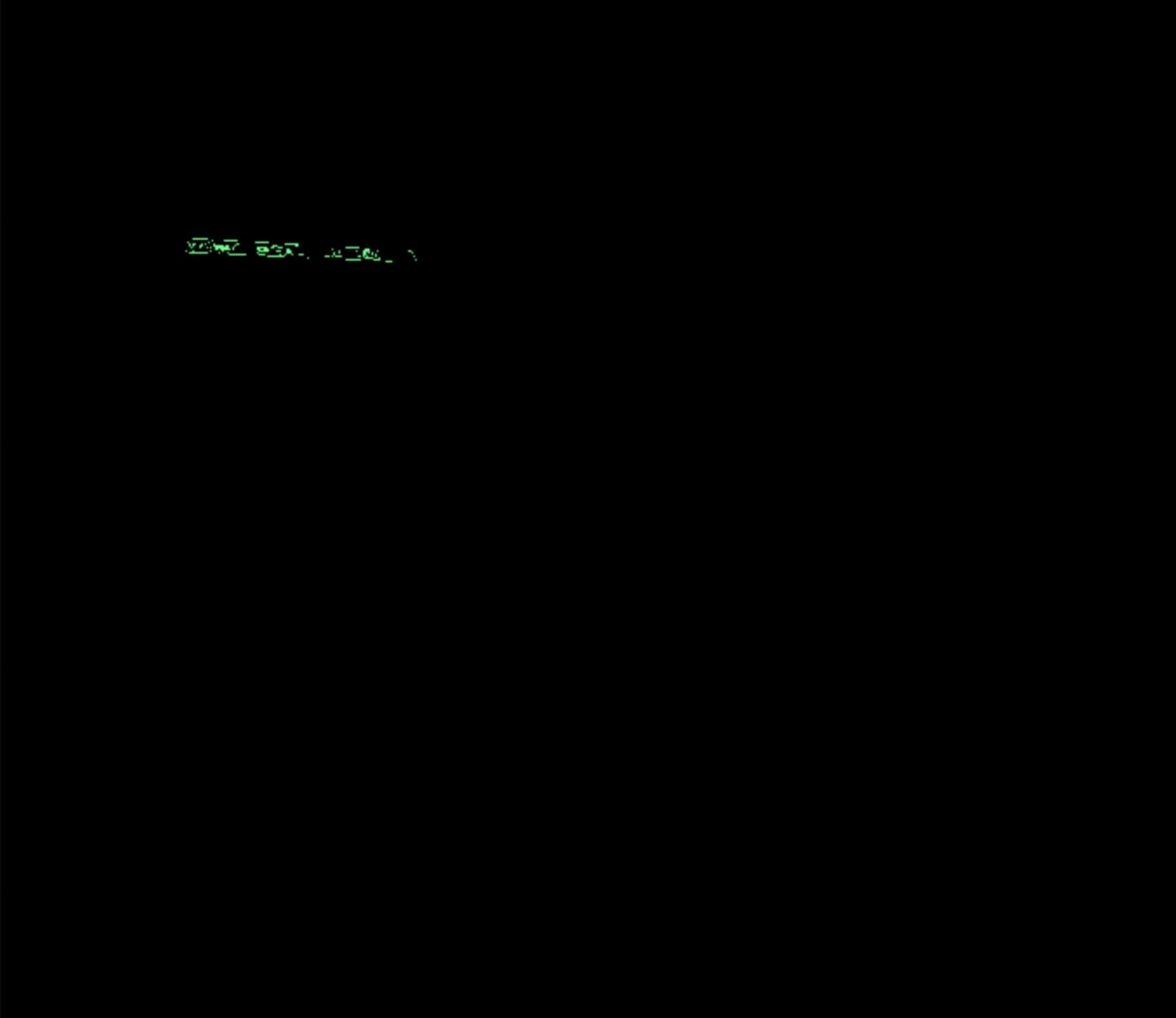


FREE Concept Car 002 02 - public domain (CC0)  
3D Model

멋진 자동차를 불러와보자

## 02 설치 방법 및 간단한 구현

---



??

- 조명 설정

```
// 조명 추가
const ambientLight = new THREE.AmbientLight(0xffffff, 0.5);
// 전체적인 빛 추가
scene.add(ambientLight);

const directionalLight = new THREE.DirectionalLight(0xffffff,
1); // 방향성 빛 추가
directionalLight.position.set(0, 1, 0); // 위에서 아래로 비추는 빛
scene.add(directionalLight);
```

### ambientLight

모든 방향에서 동일하게 빛나는 광원  
(색상, 빛의 강도)로 설정 가능

### directionalLight

특정한 방향에서 오는 빛  
position으로 x,y,z 좌표 설정 가능

## 02 설치 방법 및 간단한 구현

---



안녕하세요

000 개발자

000 개발자

조명 설정 후 제대로 출력되는 모습

- 애니메이션 설정

```
// 애니메이션 함수
const animate = () => {
  requestAnimationFrame(animate);
  renderer.render(scene, camera);
};
```

### **requestAnimationFrame()**

자바스크립트의 내장함수로써 매 프레임마다 호출되어 다음 프레임에 호출할 애니메이션을 지정하고 이를 호출하는 함수

### 엥 근데 requestAnimationFrame에 마우스 이벤트가 달려있나?

```
// OrbitControls 설정  
const controls = new OrbitControls(camera, renderer.domElement);
```

requestAnimationFrame은 직접적으로 마우스 이벤트를 처리하지 않음  
단, 이를 사용함으로써 렌더링 루프에서 지속적으로 씬을 업데이트하게 되며,  
사용자의 입력이나 다른 동적 요소들에 효과적으로 반응할 수 있다.

즉, OrbitControls와 같은 도구를 사용할 때,  
이벤트에 의해 호출된 렌더링 함수가  
requestAnimationFrame에 의해 반복적으로 호출되어,  
사용자의 조작이나 씬의 변화가 매끄럽게 화면에 반영되는 것

## 03 기업에서 요구하는 이유

**모집부문 / 상세내용**

**사용 기술**

three.js WebGL OpenGL AI/인공지능

**주요업무**

- Web 기반 2D/3D 지도 프론트엔드 서비스 개발
- 공간 데이터 기반 2D/3D 지도 제작 알고리즘 및 도구 연구/개발
- 공간 데이터 처리 및 데이터 2D/3D 가시화(Visualization) 시스템 개발
- AI 기술 연동 인터랙티브 시각화 컨텐츠 개발

**자격요건**

- Computer Graphics, 3D Geometry 이론에 대한 전반적인 이해
- 웹 기반 컴퓨터 그래픽스(WebGL, Three.js) 관련 프로젝트 개발 경험이 있는 분
- 2D/3D Canvas, SVG 등에 대한 이해 및 개발 경험

**기술스택**

JavaScript TypeScript three.js React Webpack WebRTC

HTML5 Vue.js GraphQL CSS 3

**주요업무**

- 서비스 화면 UI/UX 개발
- 인터랙티브 3D Web 구현
- Web Builder 연구 개발

**자격요건**

- 관련 경력 5년 이상
- React / Vue.js 등의 SPA framework 활용 경험
- Web 3D Graphic Library (Three.js, Babylon.js 등) 경험

현재 채용시장에서, 은근 자주 보이는 Three.js

## 03 기업에서 요구하는 이유

---

### 1. 향상된 사용자 경험 제공 가능

3D 객체가 움직이는 것 하나만으로도, 몰입감있는 경험을 선사해줌

### 2. 비주얼 마케팅

소비자에게 실제와 같은 제품 체험을 제공해줌

ex) 현대 자동차, 아이폰, 캐논 등

### 3. 메타버스 플랫폼 구축

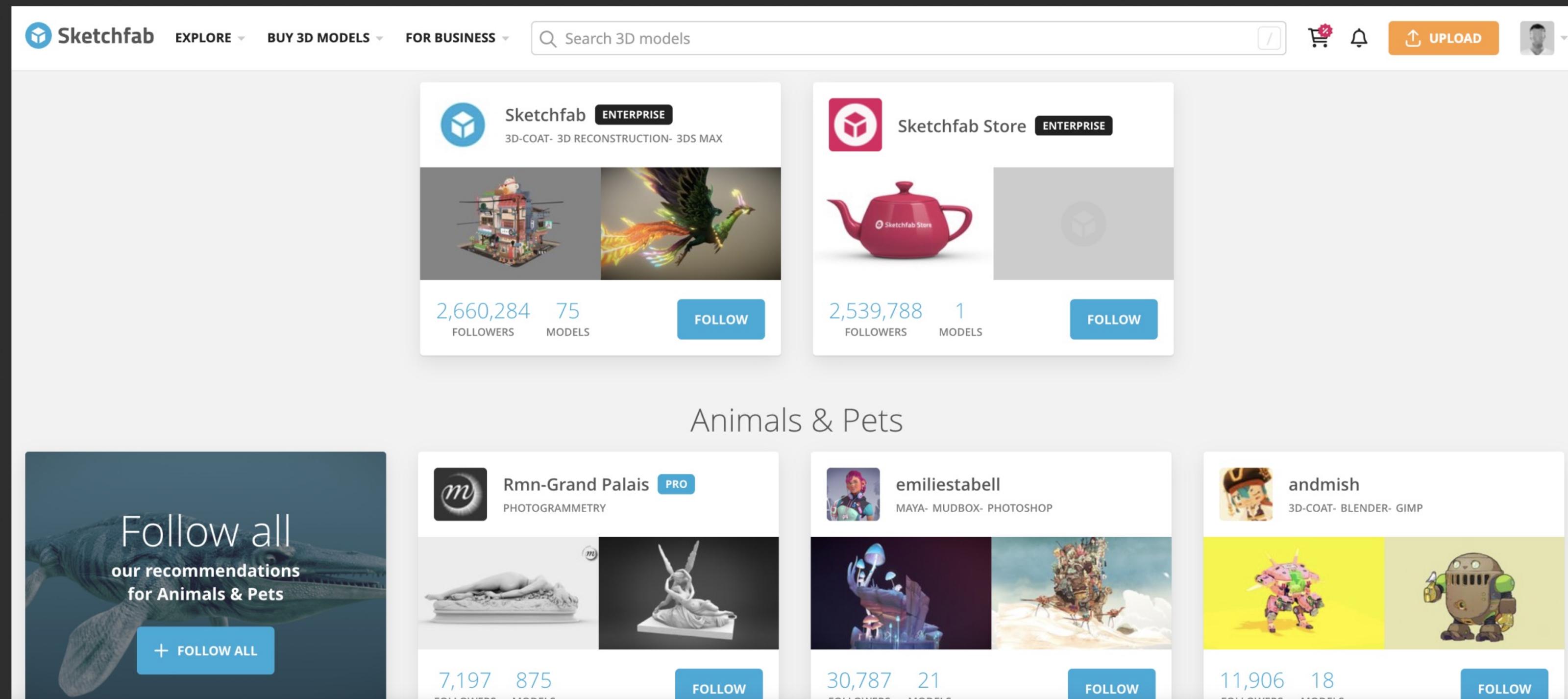
코로나19 이후, 비대면 산업 규모가 증가하고 이로인한 메타버스 플랫폼 수요가 증가

Three.js를 활용한 메타버스 구축 시도도 증가하고 있음

### 4. 가상 현실(VR) 및 증강 현실(AR) 지원

Three.js는 WebVR과 WebXR을 지원하여, 개발자들이  
복잡한 설정 없이도 VR 및 AR 환경을 구현하게 함

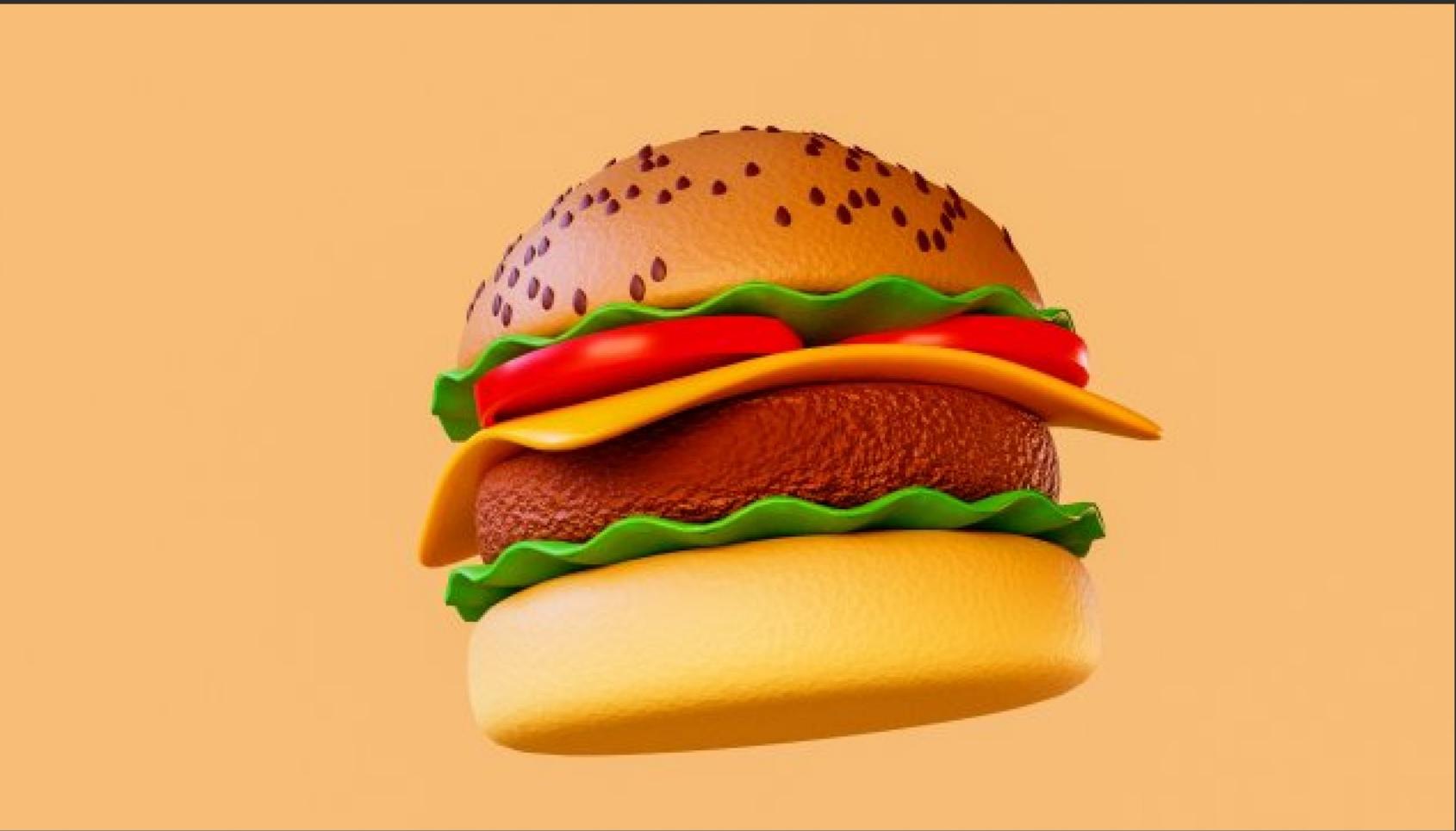
## 04 마무리



무료 3D 모델도 많으니 활용하기 좋음

## 04 마무리

---



**COMING SOON**

햄버거 커뮤니티 프로젝트에 활용할 예정