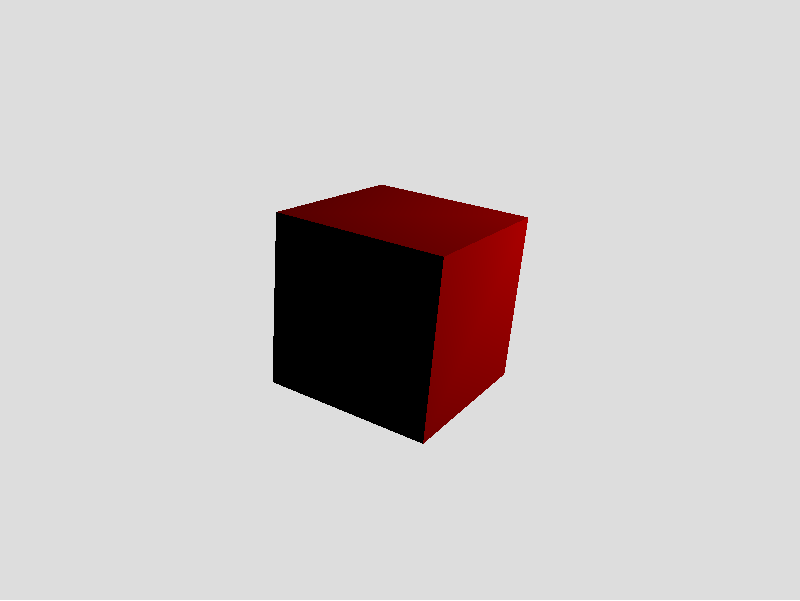
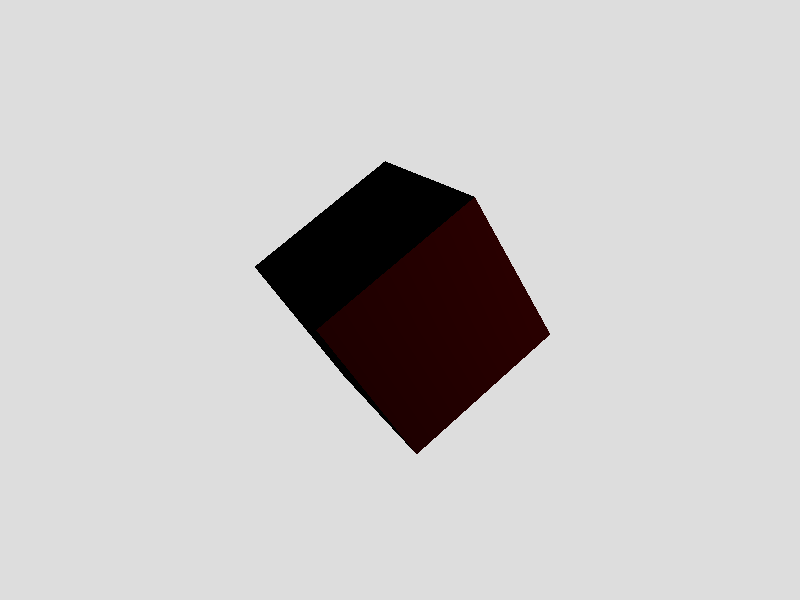
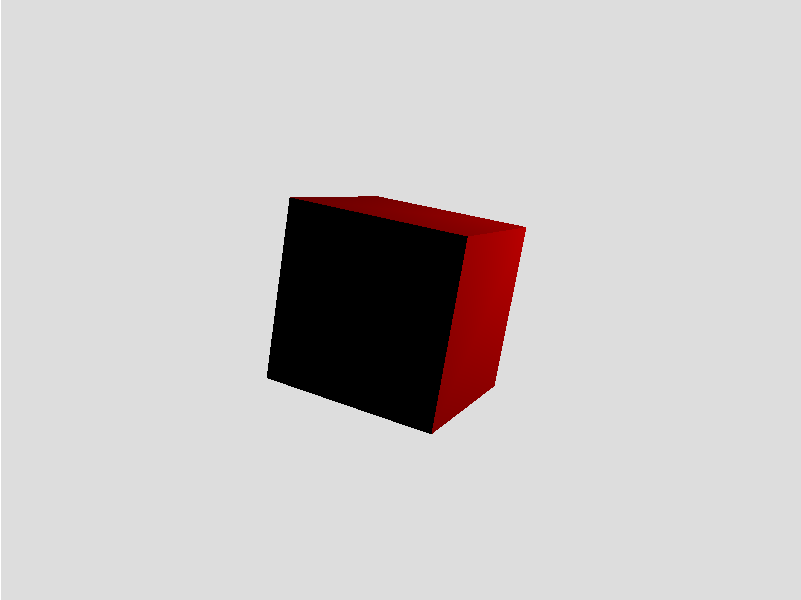
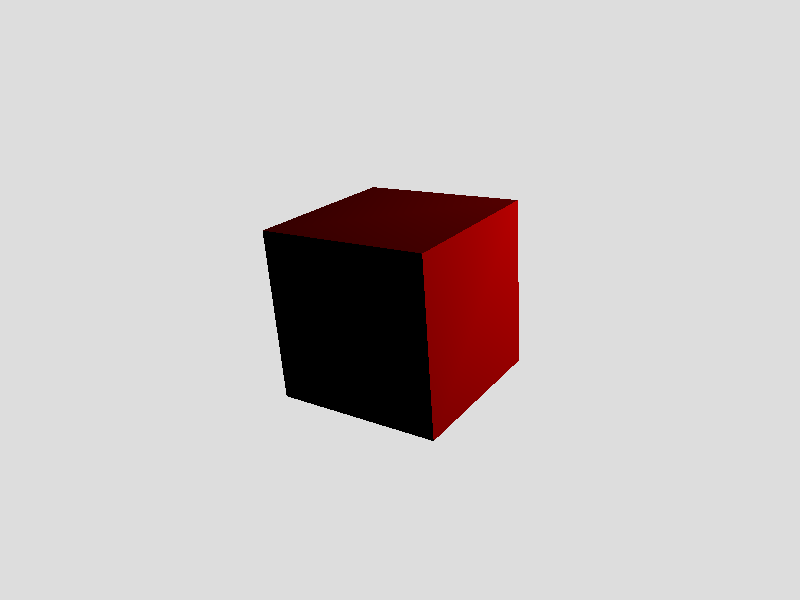
[Three.js Basic & Tutorial 1]





**[Tutorial 결과물 – 간단한 붉은 정육면체 객체 회전시키기]**

*(아주 간단한 예제이니 겁먹지 말고 잘 따라와 주세요~ ☺)*

**▶ Three.js를 시작하기 전 알아야 할 사실!**

: Web에 객체(사물)를 만들어 우리의 눈에 보이게 하려면 무엇이 필요할까요? 총 기본적인 5가지 객체들이 필요합니다. 그 중 첫번째는 객체(사물)들이 있을 화면이며 세계(world)인 “Scene”이 필요합니다. 두번째는 객체(사물)를 보는 화면 역할을 하는 “Camera”가 필요합니다. 세번째는 실제 사물의 역할을 “Mesh”라는 객체가 필요합니다. 네번째는 객체(사물)를 비추는 불빛 역할을 하는 “Light”가 필요합니다. 마지막으로 객체(사물)를 사용자 눈에 보이게 바꾸어 주는 “Renderer”가있습니다.

1. Scene : scene의 객체는 세계(world)이다. 이후에 배우겠지만 Scene객체가 세계이기 때문에 Mesh객체와 light객체 등을 Scene객체 안에 add하여 붙여 넣을 것이다.

Ex) scene.add( mesh ); , scene.add( light );

1. Camera : camera 객체는 대표적으로 PerspectiveCamera객체와 OrthographicCamera객체가 있다. 전자는 원근법을 적용하는 카메라이고, 후자는 원근법을 적용하지 않는 카메라이다.

Ex) PerspectiveCamera( )생성자 함수의 4가지 매개변수

* Angle : 시야각을 의미 (일반적으로 45도~75도정도를 많이 사용)
* Aspect : aspect ratio를 뜻하고 화면의 비율을 나타냄 (보통 width/height로 사용)
* Near : 물체를 얼마나 가까이 있는 것까지 볼것인지를 표현
* Far : 물체를 얼마나 멀리 있는 것까지 볼것인지를 표현

1. Mesh (객체) : Mesh 객체를 사용하기 위해선 Geometry객체와 Material객체가 필요하다. Geometry객체는 Mesh객체의 형태를 나타내고, Material객체는 Mesh객체의 색상을 나타낸다.

Ex) var mesh = new THREE.Mesh( geometry, material );

1. Light : light 객체는 다른 객체(사물)들이 보여지게하는 불빛을 담당하는 객체이다.

Ex) var light = new THREE.PointLight( 0xFFFF00 );

1. Renderer : Renderer 객체는 컴퓨터 안에 수식으로 맞추어진 3차원 수식을 우리가 볼 수 있는 형태로 바꾸는 기능을 합니다. Renderer객체 안에는 domElement와 같은 속성이 있으며 Renderer객체는 Canvas태그이기 때문에 Renderer객체 설정 후 Canvas태그를 화면에 추가해준다.

Ex) var renderer = new THREE.WebGLRenderer( );

renderer.serSize( 800, 600 );

document.body.appendChild( renderer.domElement );

**추가** : Renderer객체의 render( ) 메서드를 실행해야 화면에 그리므로 항상 꼭 써 줘야한다.

Ex) renderer.render(scene, camera);

**[Tutorial 전체 코드]**

1 <!DOCTYPE html>  
2 <html lang=**"en"**>  
3 <head>  
4 <meta charset=**"UTF-8"**>  
5 <title>**Three.js Tutorial**</title>  
6 <script src=**"./three.js-master/build/three.min.js"**></script>  
7 </head>  
8 <body>  
9 <script>  
10 window.onload = function() {  
11   
12 var renderer = new ***THREE***.WebGLRenderer();  
13 renderer.setSize( 800, 600 );  
14 document.body.appendChild( renderer.domElement );  
15   
16 var scene = new ***THREE***.Scene();  
17   
18 var camera = new ***THREE***.PerspectiveCamera(  
19 45, *// 시야각*20 800 / 600, *// 화면 비율*21 0.1, *// 가까이 볼 수 있는 정도*22 10000 *// 멀리 볼 수 있는 정도*23 );  
24 camera.position.set( -15, 10, 10 );  
25 camera.lookAt( scene.position );  
26   
27 var geometry = new ***THREE***.BoxGeometry( 5, 5, 5 );  
28 var material = new ***THREE***.MeshLambertMaterial( { color: 0xFF0000 } );  
29 var mesh = new ***THREE***.Mesh( geometry, material );  
30 scene.add( mesh );  
31   
32 var light = new ***THREE***.PointLight( 0xFFFF00 );  
33 light.position.set( 10, 7, 10 );  
34 scene.add( light );  
35   
36 renderer.setClearColor( 0xdddddd, 1);  
37 var *loop* = function(){  
38 *//Mesh 객체를 회전 시킵니다.*39mesh.rotation.x += 0.02;  
40 mesh.rotation.y += 0.02;  
41 mesh.rotation.z += 0.02;  
42   
43 *//랜더링을 시작합니다.*44renderer.render( scene, camera );  
45 requestAnimationFrame(*loop*);  
46 };  
47 *//게임 루프를 시작합니다.*48requestAnimationFrame(*loop*);  
49 };  
50 </script>  
51 </body>  
52 </html>

**[부분 코드 분석 및 설명]**

**Step1. body부분을 시작하기 전 기본적인 HTML코드**

1 <!DOCTYPE html>  
2 <html lang=**"en"**>  
3 <head>  
4 <meta charset=**"UTF-8"**>  
5 <title>**Three.js Tutorial**</title>  
6 <script src=**"./three.js-master/build/three.min.js"**></script>  
7 </head>

Line 1 : HTML5이라는 문서 타입을 설정하며 선언해주는 코드이다.

Line 2 : 웹 정보들을 표현하는데 사용할 언어를 설정해주는 코드이다. <html lang>은 휴먼랭귀지라는 뜻으로 랭귀지의 약자 lang을 사용하여 표현하였고, 여기서 사용된 en은 영어를 뜻하는 english를 뜻한다.

Line 3~7 : Head부분으로써 브라우저의 정보, 메타정보, 스크립트, 스타일 시트등의 요소를 포함한다. 4번째 줄은meta 정보로서 문자 표현을 유니코드로 설정하는 코드이다. 5번째 줄은 웹의 title을 정의하는 코드이며 6번째줄은 thee.js라이브러리를 현재 프로젝트에 추가하는 코드이다.

**Step2. body부분 중 Renderer객체와 Scene객체 알아보기**

8 <body>  
9 <script>  
10 window.onload = function() {  
11   
12 var renderer = new ***THREE***.WebGLRenderer();  
13 renderer.setSize( 800, 600 );  
14 document.body.appendChild( renderer.domElement );  
15   
16 var scene = new ***THREE***.Scene();

Line 8 : body부분의 시작이며 Html의 많은 내용들을 표현하는 핵심적인 부분이다.

Line 9 : script부분의 시작이며 Java script의 코드들을 표현할 수 있고, 객체 선언, 활용, 실질적인 기능적 코딩을 할 수 있는 부분이다.

Line 10 : 해당 Html 문서를 열자마자 바로 Java script로 짜여진 function( ){ … }함수 내의 내용들을 바로 실행하기 위해서 사용하는 코드이다. Window.onload = function( )을 사용하면 전체 Html문서가 완료되고 해당 function의 내용이 실행된다. 만약 head부분에 Window.onload = function( )을 사용한다면, 전체 문서가 완료되고 난 후 즉, 다시 말해서 body부분이 실행되고 난 후 head부분의 function( )의 내용이 실행되는 제어의 경로를 가지게 된다.

Line 12~14

: 첫번째로 알아볼 객체는 5가지 핵심 객체 중 Renderer객체 먼저 알아보자.

* Renderer객체로서, three.js라이브러리의 WebGLRenderer라는 <canvas>요소를 포함하는 객체를 사용하여 선언을 해준다.
* Renderer의 사이즈 (width, height)를 설정해준다.
* 만든 Renderer를 해당 Html문서의 body인 즉, 다시 말해서 Rendering을 지원할 Dom객체에 붙인다.

: 두번째로 알아볼 객체는 5가지 핵심 객체 중 Scene객체이다.

* Scene객체로서, three.js라이브러리의 Scene( )을 이용하여 선언해준다.

**Step3. body부분 중 Camera객체 알아보기**

18 var camera = new ***THREE***.PerspectiveCamera(  
19 45, *// 시야각*20800 / 600, *// 화면 비율*210.1, *// 가까이 볼 수 있는 정도*2210000 *// 멀리 볼 수 있는 정도*23 );  
24 camera.position.set( -15, 10, 10 );  
25 camera.lookAt( scene.position );

Line 18~25

: 세번째로 알아볼 객체는 5가지 핵심 객체 중 Camera객체이다.

* Camera객체로서, three.js라이브러리의 PerspectiveCamera라는 원근법을 적용해주는 카메라 객체를 사용하여 선언해준다.

(PerspectiveCamera 함수의 매개변수는 “**▶Three.js를 시작하기 전 알아야 할 사실!”**의 2번을 참고!)

* PerspectiveCamera 생성자 함수의 각 매개변수들의 수치를 설정하고 괄호를 닫는다.
* Camera의 위치를 x, y, z축을 기준으로 설정해준다.
* 생성된 camera객체가 어디를 바라보게 할 것인지를 설정하기 위해 세계(world)를 표현하는 Scene객체의 위치로 바라보게 한다.

**Step4. body부분 중 Mesh객체 알아보기**

27 var geometry = new ***THREE***.BoxGeometry( 5, 5, 5 );  
28 var material = new ***THREE***.MeshLambertMaterial( { color: 0xFF0000 } );  
29 var mesh = new ***THREE***.Mesh( geometry, material );  
30 scene.add( mesh );

Line 27~30

: 네번째로 알아볼 객체는 5가지 핵심 객체 중 Mesh객체이다.

* Mesh객체는 선언 전 2가지 보조 객체가 필요하다. geometry라는 형태를 표현해주는 객체와 material이라는 색상을 표현해주는 객체가 필요한데, 이를 27~28번째 줄에서 생성해준다. geometry객체로서 정육면체 사이즈 (5, 5, 5)짜리의 형태를 만들고, material객체로서 RGB값 기준 0xFF0000값으로 설정한 붉은색으로 색상을 표현한 객체를 만들었다.
* 29번째 줄에서 Mesh객체로서, Mesh( )라를 생성자 함수의 2가지 매개변수로 위에서 생성한 geometry와 material객체를 사용하여 선언한다.
* Mesh객체를 생성하고 가장 중요한 것은 생성된 mesh객체를 세계(world)인 scene에 붙여주는 것이다.

**Step5. body부분 중 Mesh객체 알아보기**

32 var light = new ***THREE***.PointLight( 0xFFFF00 );  
33 light.position.set( 10, 7, 10 );  
34 scene.add( light );

Line 32~34

: 마지막으로 알아볼 객체는 5가지 핵심 객체 중 Light객체이다.

* Light객체로서, three.js라이브러리의 PointLight라는 생성자 함수로 RGB값 기준 0xFFFF00인 노란색으로 빛 객체를 선언한다.
* light객체의 위치를 설정 해주기 위해 (x, y, z)기준으로 적당한 위치를 설정해준다.
* Mesh객체와 마찬가지로 Light객체를 만들었으면, 이것을 세계(world)인 scene에 붙여주어야 한다.

*(자~ step6를 하기 전 지루하시죠? 중간 결과를 한번 볼까요? ☺)*

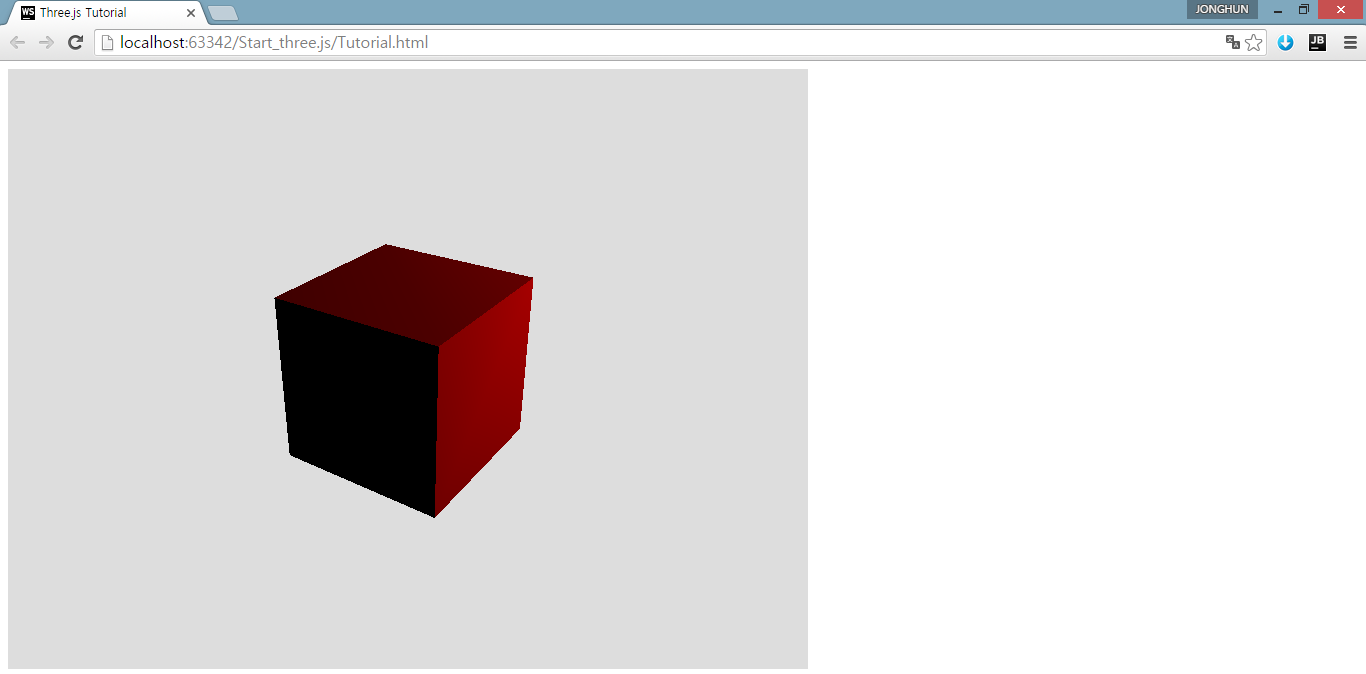
\*중간 결과 보는 방법!



1. 전체 코드 중 37~48번째 줄은 /\* … \*/를 이용하여 주석을 달아주세요.

2. 36번째 줄 뒤에 renderer.render( scene, camera );를 적어주세요.

3. 이제 Run을 시켜볼까요?



자~ 이제 Web에 붉은색 정육면체의 Mesh객체가 생성된 것을 확인 할 수 있습니다. 하지만! 너무 단순하면 재미가 없으니! 이 붉은 정육면체 객체를 회전시켜봅시다! (Continue step6~)

**Step6. Mesh객체 회전시키기 & 마무리**

36 renderer.setClearColor( 0xdddddd, 1);  
37 var *loop* = function(){  
38 *//Mesh 객체를 회전 시킵니다.*39mesh.rotation.x += 0.02;  
40 mesh.rotation.y += 0.02;  
41 mesh.rotation.z += 0.02;  
42   
43 *//랜더링을 시작합니다.*44renderer.render( scene, camera );  
45 requestAnimationFrame(*loop*);  
46 };  
47 *//게임 루프를 시작합니다.*48requestAnimationFrame(*loop*);  
49 };  
50 </script>  
51 </body>  
52 </html>

Line 36 : Renderer객체를 사용하여 기본적으로 배경이 (0x000000=검정색)으로 설정되어 있던 body부분의 색을 (0xdddddd=회색)으로 Clear해주는 코드이다.

Line 37~46

: requestAnimationFrame이라는 함수를 이용하여 Mesh객체를 회전시킬 loop함수를 재귀적으로 계속 호출시키는 코드이다.

**\*잠깐! 여기서 requestAnimationFrame함수에 대하여 조금만 더 공부해보자!**

requestAnimationFrame란?

: HTML5를 지원하는 웹브라우저 기반의 JavaScript의 Window 객체에는 requestAnimationFrame함수가 정의 되어있다. 더 정확히는, 이 함수는 WebGL의 지원과 함께 제공된다. 이 함수는 지정된 연산(함수)의 호출을 다른 연산에 방해를 주지 않고 최대한 빨리 호출해 주는 함수이다. 우리 예제와 같이 requestAnimationFrame함수를 통해 loop함수를 호출하며, loop함수의 호출을 requestAnimationFrame함수를 통해 등록해 놓으면 가장 적합하고 빠른 시점에 loop함수를 호출할 수 있게 된다.

* loop라는 이름의 function( )을 선언한다.
* 앞서 만든 Mesh객체를 x, y, z축을 기준으로 loop함수 1회 기준 0.02씩 회전시킨다.
* 이후 renderer객체를 사용해 scene과 camera를 랜더링한다.
* 다시 requestAnimationFrame함수를 통해 loop함수를 재귀호출한다.

Line 48 : requestAnimationFrame함수를 통해 loop함수를 다른 연산에 방해를 주지 않고 초기 빠른 호출을 한다.

Line 50 : script태그 종료

Line 51 : body태그 종료

Line 52 : html문서 태그 전부 종료

*(자~ 드디어 최종 결과물을 확인하실 수 있습니다! ☺)*

▶ 중간결과물은 붉은 정육면체였는데 이젠 회전하는 붉은 정육면체를 여러분이 웹에서 직접 구현하신 겁니다!



***다음 Tutorial 2에서는 조금 더 많은 기능을 사용하여 심화된 내용으로 배워봅시다~***