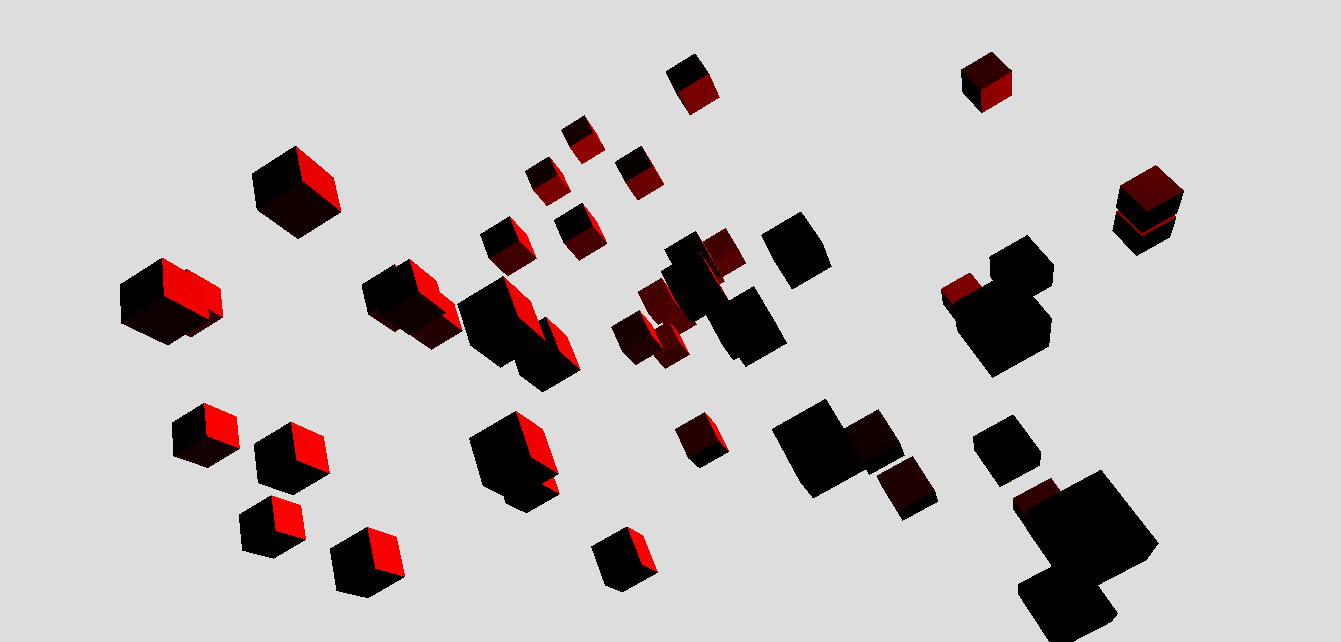
[조금 더 나아간 Tutorial1\_1]



**[Tutorial1\_1 결과물 – Tutorial1의 내용을 조금 발전시킨 내용]**

(자~ 먼저 이전 내용을 생각하며, Tutorial1의 소스를 모두 복사 해옵시다!)

**▶ Tutorial1의 소스를 그대로 가져와서 추가적인 소스코드를 작성한다.**

|  |  |
| --- | --- |
| **소스코드 구분 및 비교** | |
| **Tutorial1\_1 (오늘 내용)** | **Tutorial1 (이전 내용)** |
| 1 <!DOCTYPE html> 2 <**html** lang=**"en"**> 3 <**head**> 4 <**meta** charset=**"UTF-8"**> 5 <**title**>Three.js Tutorial1\_1</**title**> 6 <**script** src=**"./three.js-master/build/three.js"**></**script**> 7 </**head**> 8 <**body**> 9 <**script**>  10 **window**.onload = **function**() { 11 12**var** width = **window**.**innerWidth**; 13 **var** height = **window**.**innerHeight**; 14 **var** *random* = **function**(limit){ 15 **return Math**.round(**Math**.random() \* limit); 16 }; 17 **var** renderer = **new *THREE***.WebGLRenderer(); 18 renderer.setSize( width, height ); 19 **document**.**body**.appendChild( renderer.**domElement** ); 20  21 **var** scene = **new *THREE***.Scene(); 22  23 **var** camera = **new *THREE***.PerspectiveCamera( 24 45, *// 시야각* 25 width / height, *// 화면 비율* 260.1, *// 가까이 볼 수 있는 정도* 27 10000 *// 멀리 볼 수 있는 정도* 28); 29 camera.**position**.set( -15, 10, 10 ); 30 camera.lookAt( scene.**position** ); 31  32 *//객체를 생성합니다.* 33 ***list*** = []; 34 **for** (**var** i = 0; i < 50; i++){ 35 **var** geometry = **new *THREE***.CubeGeometry(1,1,1); 36 **var** material = **new *THREE***.MeshLambertMaterial({**color**: 0xFF0000}); 37 *//Mesh객체를 생성합니다.* 38 ***mesh*** = **new *THREE***.Mesh(geometry, material); 39 **mesh**.**position**.set(*random*(20) - 10, *random*(10) - 5, *random*(20) - 10); 40 scene.add(**mesh**); 41 **list**.push(**mesh**); 42 } 43  44 **var** light = **new *THREE***.PointLight( 0xFFFF00 ); 45 light.**position**.set( 10, 7, 10 ); 46 scene.add( light ); 47  48 **var** *loop* = **function**(){ 49 **list**.forEach(**function**(item){ 50 *//애니메이션을 수행합니다.* 51 item.**rotation**.**x** += 0.02; 52 item.**rotation**.**y** += 0.02; 53 item.**rotation**.**z** += 0.02; 54 }); 55  56 *//랜더링을 시작합니다.* 57renderer.render( scene, camera ); 58 requestAnimationFrame(*loop*); 59 }; 60 renderer.setClearColor( 0xdddddd, 1); 61 *//루프를 시작합니다.* 62requestAnimationFrame(*loop*); 63 }; 64 </**script**> 65 </**body**> 66 </**html**> | 1 <!DOCTYPE html> 2 <**html** lang=**"en"**> 3 <**head**> 4 <**meta** charset=**"UTF-8"**> 5 <**title**>Three.js Tutorial1</**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* 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** ); 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 *//루프를 시작합니다.* 48requestAnimationFrame(*loop*); 49 }; 50 </**script**> 51 </**body**> 52 </**html**> |

**[추가된 코드 집중 분석]**

|  |  |
| --- | --- |
| **설명** | **추가된 Tutorial1\_1 소스** |
| **Part1.**  Line 12~13 : width, height변수를 화면의 크기에 맞는 사이즈로 값을 초기화하여 선언한다.  (앞으로 너비와 높이값이 필요한 자리에 모두 width와 height변수를 사용할 것이다.)  Line 14~16 : random함수를 만들었다. return값으로 라이브러리의 Marh함수를 사용하여 난수 값에 매개변수를 곱한 값을 설정하였다. | 12**var** width = **window**.**innerWidth**; 13 **var** height = **window**.**innerHeight**; 14 **var** *random* = **function**(limit){ 15 **return Math**.round(**Math**.random() \* limit); 16 }; |
| **Part2.**  Line 32~33 : list라는 배열을 만들었다.  (이 list배열을 사용하여 많은 서로 다른 객체인 Mesh들을 저장 할 것이다.)  Line 34 : 50회동안 반복하는 구문을 사용  Line 35~36 : mesh에 사용할 geometry와 material 객체를 선언한다.  Line 38 : mesh객체를 geometry와 material 객체를 사용하여 생성한다.  Line 39 : 방금 생성된 mesh객체의 위치를 앞서 만든 random함수를 사용하여 x, y, z축으로 설정해준다.  Line 40 : 생성된 mesh객체를 scene에 붙인다.  Line 41 : 붙여진 mesh객체를 list배열에 push한다. (배열관리)   * *이 과정을 50회 반복하여 랜덤한 위치에 50개의 붉은 직사각형이 만들어 진다.* | 32 *//객체를 생성합니다.* 33 ***list*** = []; 34 **for** (**var** i = 0; i < 50; i++){ 35 **var** geometry = **new *THREE***.CubeGeometry(1,1,1); 36 **var** material = **new *THREE***.MeshLambertMaterial({**color**: 0xFF0000}); 37 *//Mesh객체를 생성합니다.* 38 ***mesh*** = **new *THREE***.Mesh(geometry, material); 39 **mesh**.**position**.set(*random*(20) - 10, *random*(10) - 5, *random*(20) - 10); 40 scene.add(**mesh**); 41 **list**.push(**mesh**); 42 } |
| **Part3.**  Line 48 : 이전 예제에서 사용했던 loop함수를 다시 사용한다.  Line 49~54 : list배열의 forEach함수를 사용하여 배열값들을 매개변수로 각 mesh들의 x, y, z축을 0.02씩 회전시킨다.   * list배열 속의 모든 mesh들이 회전하는 효과를 준다.   Line 57 : renderer를 사용하여 scene과 camera를 랜더링한다.  Line 58 : requestAnimationFrame함수를 사용하여 loop함수를 호출해준다. | 48 **var** *loop* = **function**(){ 49 **list**.forEach(**function**(item){ 50 *//애니메이션을 수행합니다.* 51 item.**rotation**.**x** += 0.02; 52 item.**rotation**.**y** += 0.02; 53 item.**rotation**.**z** += 0.02; 54 }); 55  56 *//랜더링을 시작합니다.* 57renderer.render( scene, camera ); 58 requestAnimationFrame(*loop*); 59 }; |

(자~ 오늘도 최종 결과물을 확인해 봅시다!)

▶ Tutorial1의 결과물에서 코드를 조금 추가하여 심화된 내용으로 예제를 진행해 보았습니다.

* Tutorial1\_1 결과물 링크 : <https://vimeo.com/152284674>

***다음 Tutorial 2에서는 색다른 예제로 배워봅시다~***