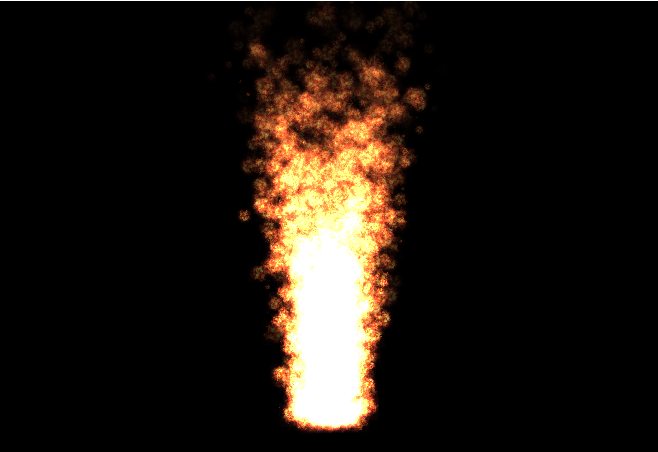
[Tutorial 2: 타오르는 불 만들어보기!]



**[Tutorial 2의 결과물 – 타오르는 불 객체 만들어보기!]**

*(Tutorial 2부터는 실제로 많이 사용되는 예제들로 준비해봤습니다! 잘 따라와주세요! ~ ☺)*

**▶ 타오르는 불 객체를 만들기 위해서 다음과 같은 순서로 진행을 할 것입니다.**

1. 필요한 js라이브러리 연결시키기
2. Three.js의 기본적인 main객체들 생성하기
3. 핵심GUI객체 만들고 준비하기
4. Rendering과 Animate하여 불 구현하기
5. 만들어진 타오르는 불 객체 확인하기

*(전체적으로 소스코드를 점점 붙여 나가면서 완성됩니다. 각 Step별로 추가된 소스코드에 집중해주세요!)*

**Step1. 필요한 js라이브러리 연결시키기**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 이번 Tutorial 2예제를 수행하기 위하여 필요한 라이브러리는 총 2가지이다.   1. three.min.js 2. ParticleSystem.js   Three.js라이브러리를 사용하기위하여 Tutorial1에서도 사용하였던 three.min.js와 추가적인 1가지가 있다. 불 요소를 표현하기위하여 사용할 ParticleSystem.js이 있다.  <코드 1>의 7~8번째줄의 부분이 상대경로를 이용한 js라이브러리를 연결시키는 부분이다.  이번 타오르는 불 예제는 위에서 연결시킨 라이브러리에서 많은 함수들을 구현해 놨기 때문에 메인 HTML문서에서는 라이브러리의 기능을 사용하여 쉽게 불을 구현할 수 있다. | |  |  |  | | --- | --- | --- | | 1  2  3  4  5  6  7  8  9  10  11 | <!DOCTYPE html>  <html lang="en">  <head>      <meta charset="UTF-8">      <title>Three.js Tutorial2</title>        <script src="./three.js-master/build/three.min.js"></script>      <script src="./ParticleSystem.js"></script>    </head>  <body> |  | |

**<코드1>**

**Step2. Three.js의 기본적인 main객체들 생성하기 & 핵심 GUI객체 만들고 준비하기**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| <코드 2>의 14~25번째 줄까지는 앞으로 타오르는 불 객체를 생성하기 위하여 필요한 변수들을 전역으로 생성하여 주는 코드이다.  Line 14 : clock변수는 Three.js라이브러리의 THREE.Clock( )을 사용하여 생성하였다. 이 clock변수는 이후 getDelta( )라는 함수를 사용하여 재귀호출시 경과한 시간을 가져오는 역할을 할것이다.  Line 15 : Tutorial 1에서 배웠던 scene, renderer, camera와 같은 main변수들을 선언하였다.  Line 16 : particlaSystem이라는 이번 Tutorial 2에서 사용할 핵심 변수를 선언하였다.  Line 17~25 : gui와 ParamConfig변수를 선언하는 코드이다. 이는 불 객체의 필요요소들을 정의하는데 필요한 변수들이다. 앞으로 배울 Method에서 중요하게 사용된다.   * seedVelDir는 불 객체의 위치를 나타낸다. * seedVelMag는 불이 타오르는 정도를 나타낸다. * seedLife는 불이 생성되고 난 후 소멸될때까지의 생명주기를 나타낸다. * seedSize는 불의 작은 조각의 사이즈를 나타낸다. * seedSpread는 불의 타오를 때 퍼지는 정도를 나타낸다. * colorBase는 기본 색상을 나타낸다. * 마지막줄의 tex는 Texure의 줄임말로 불로 사용할 객체의 작은 조각 텍스쳐를 load하는 역할을 한다.   이러한 전역변수의 선언이 끝나고 본격적으로 중요 함수들에 대하여 알아보자! | |  |  |  | | --- | --- | --- | | 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26 | <!DOCTYPE html>  <html lang="en">  <head>      <meta charset="UTF-8">      <title>Three.js Tutorial2</title>        <script src="./three.js-master/build/three.min.js"></script>      <script src="./ParticleSystem.js"></script>    </head>  <body>  <script>        var clock = new THREE.Clock();      var scene, renderer, camera;      var particleSystem;      var gui, ParamConfig = {          seedVelDir : new THREE.Vector3(0.7, -2, 0),          /\*seedVelMag : 500.0,          seedLife : 1000,          seedSize : 200,          seedSpread : 0.15,\*/          colorBase : new THREE.Color(0Xfffafa),          tex : THREE.ImageUtils.loadTexture("./textures/snowflake.png")      }; |  | |

**<코드2>**

**Step3. Rendering과 Animate하여 불 구현하기**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| <코드 3>의 27~56번째 줄 까지는 init( )함수에 관한 코드이다.  Line 29~31 : Renderer객체를 생성하고 size와 clearclolor를 설정한다.  Line 33 : 생성한 Renderer객체를 자식 노드로서 붙인다.  Line 35~36 : Camera객체를 생성하고 position을 설정한다.  Line 38 : Scene객체를 생성한다.  Line 40~41 : ParticleSystem.js에 있는 particleSystem객체를 생성하고 초기화한다.  Line 43 : 생성한 particleSystem에 있는 Mesh객체를 가져와 Scene에 붙인다.  Line 45~55 : velMag, life, seed, spread변수를 선언하고 각 적절한 수치로 초기화한다. 이후 particleSystem의 setParameters를 이용하여 파티클 요소로서 설정한다.  <코드 3>의 58~73번째 줄 까지는 파티클을 구현하는 update( )함수에 관한 코드이다.  Line 60~62 : 복제된 파티클들의 방향성계산에 필요한 두 변수 center와 normal에 대한 변수로서 cen과 nor을 선언한다. 또한 radian계산에 필요한 rad변수를 선언한다. 이후 각 방향성에 맞는 수치로 초기화 한다.  Line 64 : particleSystem의 addParticleFromDisk( )함수를 사용하여 첫 매개변수인 num의 수치인 50만큼의 파티클 복제를 하여 불꽃 파티클을 구현한다.  Line 66 : particleSystem의 updateParticles를 사용하여 매개변수로서 clock.getDelta( )함수를 사용하여 이전의 이 getDelta( )함수 호출 후 경과시간을 초단위로 반환하여 updataParticles함수가 받아서 Particle들에 관한 정보를 계산하여 출력한다.  Line 68~70 : Renderer객체를 clear하고, Depth도 claer한다. 그리고 다시 renderer객체에 scene과 camera객체를 랜더링한다.  Line 72 : animate함수를 Frame이 변환될때마다 콜백함수로서 호출하기 위하여 재귀호출 하는 부분이다.  Line 75~76 : init( )함수와 animate( )함수를 호출한다.  이제 모든 코드에 대한 분석을 마쳤다. 브라우저 실행 버튼을 클릭하여 결과물을 확인해 보자! | |  |  |  | | --- | --- | --- | | 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80 | <!DOCTYPE html>  <html lang="en">  <head>      <meta charset="UTF-8">      <title>Three.js Tutorial2</title>        <script src="./three.js-master/build/three.min.js"></script>      <script src="./ParticleSystem.js"></script>    </head>  <body>  <script>        var clock = new THREE.Clock();      var scene, renderer, camera;      var particleSystem;      var gui, ParamConfig = {          seedVelDir : new THREE.Vector3(0.7, -2, 0),          /\*seedVelMag : 500.0,          seedLife : 1000,          seedSize : 200,          seedSpread : 0.15,\*/          colorBase : new THREE.Color(0Xfffafa),          tex : THREE.ImageUtils.loadTexture("./textures/snowflake.png")      };        function init() {            renderer = new THREE.WebGLRenderer();          renderer.setSize(window.innerWidth, window.innerHeight);          renderer.setClearColor(0x000000);            document.body.appendChild(renderer.domElement);            camera = new THREE.PerspectiveCamera(60, window.innerWidth / window.innerHeight, 0.1, 100000 );          camera.position.z = 2000;            scene = new THREE.Scene();            particleSystem = new ParticleSystem();          particleSystem.initialize(100000);            scene.add(particleSystem.getMesh());            var velMag = {seedVelMag : 500};          particleSystem.setParameters(velMag);            var life = {seedLife : 3};          particleSystem.setParameters(life);            var seed = {seedSize : 200};          particleSystem.setParameters(seed);            var spread = {seedSpread : 0.25};          particleSystem.setParameters(spread);      }        function animate() {            var cen = new THREE.Vector3(0,-300,0);          var nor = new THREE.Vector3(0,1,0);          var rad = 150;            particleSystem.addParticlesFromDisk(50, cen, nor, rad);            particleSystem.updateParticles(clock.getDelta());            renderer.clear();          renderer.clearDepth();          renderer.render( scene, camera );            requestAnimationFrame( animate );      }        init();      animate();    </script>  </body>  </html> |  | |

**<코드3>**

(자~ 오늘은 조금 더 심화된 결과를 기대하며 확인해 봅시다!)

▶ Tutorial2의 결과물 확인 링크입니다. 다음 예제에서는 Tutorial2\_1로서 조금 추가된 기능을 배우게 됩니다.

* Tutorial2결과물 링크 : <https://vimeo.com/153871678>

***다음 Tutorial 2\_1에서 추가 기능을 배워봅시다~***