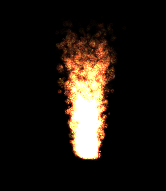
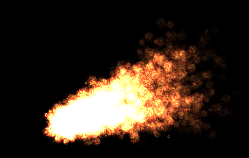
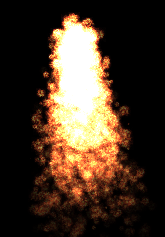
[기능이 추가된Tutorial2\_1]

**[Tutorial2\_1 결과물 – 추가된 기능 : Mouse Control]**

(자~ 먼저 이전 내용을 생각하며, Tutorial2의 소스를 모두 복사 해옵시다!)

**▶ Tutorial2의 소스를 그대로 가져와서 추가적인 소스코드를 작성한다.**

|  |  |
| --- | --- |
| **소스코드 구분 및 비교** | |
| **Tutorial2\_1 (오늘 내용)** | **Tutorial2 (이전 내용)** |
| |  |  |  | | --- | --- | --- | | 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  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109  110  111 | <!DOCTYPE html>  <html lang="en">  <head>      <meta charset="UTF-8">      <title> Three.js Tutorial2\_1</title>        <script src="./three.js-master/build/three.min.js"></script>      <script src="./ParticleSystem.js"></script>      <script src="./three.js-master/examples/js/controls/TrackballControls.js"></script>    </head>  <body>  <script>        var clock = new THREE.Clock();        var scene, renderer, camera;      var controls; // 마우스로 화면을 돌려볼수 있는 기능        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;            controls = new THREE.TrackballControls( camera );             controls.rotateSpeed = 10.0;           controls.zoomSpeed = 5;           controls.panSpeed = 2;             controls.noZoom = false;           controls.noPan = false;             controls.staticMoving = true;           controls.dynamicDampingFactor = 0.3;            scene = new THREE.Scene();            window.addEventListener( 'resize', onWindowResize, false );            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 onWindowResize() {         camera.aspect = window.innerWidth / window.innerHeight;       camera.updateProjectionMatrix();         renderer.setSize( window.innerWidth, window.innerHeight );         controls.handleResize();       }        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());            controls.update();            renderer.clear();          renderer.clearDepth();          renderer.render( scene, camera );            requestAnimationFrame( animate );      }        init();      animate();    </script>  </body>  </html> |  | | |  |  |  | | --- | --- | --- | | 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> |  | |

**[추가된 코드 집중 분석]**

|  |  |
| --- | --- |
| **설명** | **추가된 Tutorial2\_1 소스** |
| **Part1. 라이브러리 추가**  **: Mouse Control을 하기 위한 라이브러리로서 TrackballControls.js 라이브러리를 추가한다.** | **9 <script src="./three.js-master/examples/js/controls/TrackballControls.js"></script>** |
| **Part2. controls변수 선언**  **: Mouse Control을 하기 위한 controls변수 선언** | **18 var controls; // 마우스로 화면을 돌려볼수 있는 기능** |
| **Part3. Init( )함수에서의 초기화**  **: controls변수를 새로 추가한 TrackballControls라이브러리를 사용하여 객체 선언을 한다. 그리고 rotateSpeed, zoomSpeed, panSpeed, noZoom, noPan, staticMoving 그리고 dynamicDampingFactor와 같은 요소들은 초기화 한다.** | **43 controls = new THREE.TrackballControls( camera );**  **44**  **45  controls.rotateSpeed = 10.0;**  **46  controls.zoomSpeed = 5;**  **47  controls.panSpeed = 2;**  **48**  **49 controls.noZoom = false;**  **50 controls.noPan = false;**  **51**  **52 controls.staticMoving = true;**  **53 controls.dynamicDampingFactor = 0.3;** |
| **Part4. 윈도우 resize시 이벤트 리스너를 설정**  **: html이벤트로서 윈도우(window)나 프레임(frame)크기가 변경되는 경우 window resize를 하게 만드는 이벤트가 일어남** | **57 window.addEventListener( 'resize', onWindowResize, false );** |
| **Part5. onWindowResize( )이벤트 함수 구현**  **: part4에서 사용하는 리스너 실행을 위하여 onWindowResize( )함수를 구현한다. 화면 비율로서 window의 width와 height의 바뀐 비율만큼 설정하고, 프레임의 변화가 있었기 때문에 Camera Project Matrix를 update해준다. 또한 renderer또한 다시 size를 화면에 맞게 set해준다. controls객체 또한 resize를 적용시켜준다.** | **77 function onWindowResize() {**  **78**  **79 camera.aspect = window.innerWidth / window.innerHeight;**  **80 camera.updateProjectionMatrix();**  **81**  **82 renderer.setSize( window.innerWidth, window.innerHeight );**  **83**  **84 controls.handleResize();**  **85 }** |
| **Part6. controls객체의 활성화를 시킨다.**  **: animate( )함수내에서 만들어 놓은 controls객체를 update하여 활성화 시킨다.** | **97 controls.update();** |

(자~ 오늘도 최종 결과물을 확인해 봅시다!)

▶ Tutorial2의 결과물에서 코드를 조금 추가하여 심화된 내용으로 예제를 진행해 보았습니다.

* Tutorial2\_1 결과물 링크 : <https://vimeo.com/154019215>

***다음 번에는 실전 예제로 배워봅시다 ~ !***