바람에 흩날리는 낙엽 예제 따라 하기

계정 비쥬얼 시뮬레이션 연구실

바람에 흩날리는 낙엽을 보여주는 것이 이번 예제의 목표이다.   
이번 예제는 다음과 같은 순서로 진행할 것이다.

1. 필요한 js파일 연결하기.
2. Three.js의 기본적인 요소들 만들기.
3. 지면 생성하기.
4. 낙엽을 생성하기.
5. 낙엽이 쌓이는 효과주기.

**Step 1. 필요한 js파일 연결하기**

|  |  |
| --- | --- |
| <code1> | |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | <!DOCTYPE html>  <html lang="en">  <head>      <title>Leaf Terrain</title>      <meta charset="utf-8">  </head>  <body>  <script src="./lib/three/build/three.min.js"></script>  <script src="./lib/three/examples/js/controls/OrbitControls.js"></script>  <script src="./ParticleSystem.js"></script>  <script src="./AdditiveTerrain.js"></script>  </body>  </html> |

이번 예제에서 사용하는 js파일은 위의 <code1>에서 보는 것과 같은 4가지이다. 각각 기본적인 Three.js의 요소를 만들기 위해서, OrbitControls를 사용하기 위해서, 파티클 시스템을 사용하기 위해서, AddtibeTerrain을 사용하기 위해서 사용한다.

**Step 2. Three.js의 기본적인 요소들 만들기**

|  |  |
| --- | --- |
| <code2> | |
| 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 | <!DOCTYPE html>  <html lang="en">  <head>      <title>Leaf Terrain</title>      <meta charset="utf-8">  </head>  <body>  <script src="./lib/three/build/three.min.js"></script>  <script src="./lib/three/examples/js/controls/OrbitControls.js"></script>  <script src="./ParticleSystem.js"></script>  <script src="./AdditiveTerrain.js"></script>  <script>      var container;      var camera, scene, renderer, controls;        init();        function init() {            container = document.createElement( 'div' );          document.body.appendChild( container );            camera = new THREE.PerspectiveCamera( 60, window.innerWidth / window.innerHeight, 0.01, 10000 );          camera.setLens(18);            camera.position.y = 200;          camera.position.x = 200;          camera.position.z = 500;            scene = new THREE.Scene();            renderer = new THREE.WebGLRenderer( { antialias: true } );          renderer.setClearColor( 0xaaaaaa );          renderer.setSize( window.innerWidth, window.innerHeight );          container.appendChild(renderer.domElement);            controls = new THREE.OrbitControls(camera, renderer.domElement);            window.addEventListener( 'resize', onWindowResize, false );      }        function onWindowResize() {          camera.aspect = window.innerWidth / window.innerHeight;          camera.updateProjectionMatrix();            renderer.setSize( window.innerWidth, window.innerHeight );      }  </script>  </body>  </html> |

위의 <code2>와 같이 기본적인 요소들을 설정한다. 13번째 줄과 14번째 줄에서 다양한 함수에서 쓰이는 변수들은 전역변수로 선언했다. 선언한 변수들은 18번째 줄에 나타나 있는 init함수에서 초기화 했다.

**Step 3. 지면 생성하기**

|  |  |
| --- | --- |
| <code3> | |
| 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>      <title>Leaf Terrain</title>      <meta charset="utf-8">  </head>  <body>  <script src="./lib/three/build/three.min.js"></script>  <script src="./lib/three/examples/js/controls/OrbitControls.js"></script>  <script src="./ParticleSystem.js"></script>  <script src="./AdditiveTerrain.js"></script>  <script>      var container;      var camera, scene, renderer, controls;        var terrain = new AdditiveTerrain();        var min = new THREE.Vector3(-900.0, 0.0, -900.0);      var max = new THREE.Vector3( 900.0, 0.0,  900.0);      var iSize = 100;      var jSize = 100;      var totalNum = 10000;      var ptSize = 20;      var texPath = "./textures/grass.jpg";        var terrainParams = {terrainImage: texPath};      terrain.initialize(min, max, iSize, jSize, totalNum, ptSize, terrainParams);        init();      loop();        function init() {            container = document.createElement( 'div' );          document.body.appendChild( container );            camera = new THREE.PerspectiveCamera( 60, window.innerWidth / window.innerHeight, 0.01, 10000 );          camera.setLens(18);            camera.position.y = 200;          camera.position.x = 200;          camera.position.z = 500;            scene = new THREE.Scene();            scene.add(terrain.getMesh());            renderer = new THREE.WebGLRenderer( { antialias: true } );          renderer.setClearColor( 0xaaaaaa );          renderer.setSize( window.innerWidth, window.innerHeight );          container.appendChild(renderer.domElement);            controls = new THREE.OrbitControls(camera, renderer.domElement);            window.addEventListener( 'resize', onWindowResize, false );      }        function onWindowResize() {          camera.aspect = window.innerWidth / window.innerHeight;          camera.updateProjectionMatrix();            renderer.setSize( window.innerWidth, window.innerHeight );      }        var fps = 40;        function loop() {          setTimeout(loop, 1000/fps);            controls.update();            requestAnimationFrame(render);      }        function render() {          renderer.render( scene, camera );      }  </script>  </body>  </html> |

위의 <code3>의 16번째 줄에서 27번째 줄에 나온 것과 같이 지면을 생성할 수 있다. Min과 max를 만들어서 지면의 크기를 설정하고, iSize와 jSize를 사용해서 지면에 입히는 텍스처의 비율을 설정할 수 있다. totalNum은 지면에 쌓이는 낙엽의 개수를 나타내고, ptSIze는 지면에 쌓이는 낙엽의 크기를 나타낸다. terrainParams에 JSON 형식으로 지면의 텍스처를 설정할 수 있다. 이 변수들을 이용해서 initialize함수를 이용해서 지면을 만들고 원하는 데로 설정할 수 있다. 만든 지면을 init함수에서 장면에 추가하면 지면을 사용할 준비가 다 됬다.

다음으로 지면이 잘 설정됐는지 확인하기 위해서 loop함수를 만들고 render함수를 부르도록 했다. 이제 <code3>을 브라우저에서 실행시켜보면 다음의 <그림1>과 같은 지면을 확인할 수 있다.

|  |
| --- |
|  |
| <그림1> |

**Step 4. 낙엽을 생성하기**

|  |  |
| --- | --- |
| <code4> | |
| 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 | <!DOCTYPE html>  <html lang="en">  <head>      <title>Leaf Terrain</title>      <meta charset="utf-8">  </head>  <body>  <script src="./lib/three/build/three.min.js"></script>  <script src="./lib/three/examples/js/controls/OrbitControls.js"></script>  <script src="./ParticleSystem.js"></script>  <script src="./AdditiveTerrain.js"></script>  <script>      var container;      var camera, scene, renderer, controls;        var terrain = new AdditiveTerrain();        var min = new THREE.Vector3(-900.0, 0.0, -900.0);      var max = new THREE.Vector3( 900.0, 0.0,  900.0);      var iSize = 100;      var jSize = 100;      var totalNum = 10000;      var ptSize = 20;      var texPath = "./textures/grass.jpg";        var terrainParams = {terrainImage: texPath};      terrain.initialize(min, max, iSize, jSize, totalNum, ptSize, terrainParams);        var particleSystem = new ParticleSystem();      var psParams = {blendingType : THREE.NormalBlending};      particleSystem.initialize(1000, psParams);        particleSystem.setParameters({          seedVelDir: new THREE.Vector3(0.0,-1,0),          seedVelMag: 120.0,          globalForce: new THREE.Vector3(0, -0.1, 0),          windStrength: 10,          seedSize: 60,          seedLife: 5.0,          texFile: "./textures/leaf.png",          particleColor: new THREE.Color(0Xaaaa00),          angularVel : 0.005,          alpha: 1.0,      });        init();      loop();        function init() {            container = document.createElement( 'div' );          document.body.appendChild( container );            camera = new THREE.PerspectiveCamera( 60, window.innerWidth / window.innerHeight, 0.01, 10000 );          camera.setLens(18);            camera.position.y = 200;          camera.position.x = 200;          camera.position.z = 500;            scene = new THREE.Scene();            scene.add(terrain.getMesh());          scene.add(particleSystem.getMesh());            renderer = new THREE.WebGLRenderer( { antialias: true } );          renderer.setClearColor( 0xaaaaaa );          renderer.setSize( window.innerWidth, window.innerHeight );          container.appendChild(renderer.domElement);            controls = new THREE.OrbitControls(camera, renderer.domElement);            window.addEventListener( 'resize', onWindowResize, false );      }        function onWindowResize() {          camera.aspect = window.innerWidth / window.innerHeight;          camera.updateProjectionMatrix();            renderer.setSize( window.innerWidth, window.innerHeight );      }        var fps = 40;      var time;        function loop() {          setTimeout(loop, 1000/fps);            var now = new Date().getTime();          var dt = (now - (time||now))/1000;          time = now;            var cen = new THREE.Vector3(0,500,0);          var nor = new THREE.Vector3(0,-1,0);          var rad = 500;            particleSystem.addParticlesFromDisk(1, cen, nor, rad);          particleSystem.updateParticles(dt, terrain);            controls.update();            requestAnimationFrame(render);      }        function render() {          renderer.render( scene, camera );      }  </script>  </body>  </html> |

낙엽을 나타낼 파티클 시스템을 만들고 설정하기 위해서 <code4>의 29번쨰 줄에서 44번째 줄에 있는 과정을 따라가면 된다. 30번째줄이 나타내는 것은 낙엽의 불투명하게 보이게 하기 위해서 설정한 것이다. 그리고 42번째 줄의 angularVel을 사용해서 낙엽이 회전하는 모습을 그려줄 수 있다. 낙엽은 불투명한 물체이기 때문에 alpha값을 1로 맞춘다. 그리고 init화면에서 장면에 particleSystem을 추가한다.

낙엽을 내리게 하기 위해서 loop함수에서 addParticlesFromDisk함수를 사용한다. 그리고 updateParticles함수를 사용해서 시간의 지남에 따라서 낙엽의 상태를 update해준다.

<code4>를 브라우저에서 실행시키면 아래의 <그림2>와 같은 모습이 보일 것이다.

|  |
| --- |
|  |
| <그림2> |

**Step 5. 낙엽이 쌓이는 효과 주기**

|  |  |
| --- | --- |
| <code5> | |
| 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  112 | <!DOCTYPE html>  <html lang="en">  <head>      <title>Leaf Terrain</title>      <meta charset="utf-8">  </head>  <body>  <script src="./lib/three/build/three.min.js"></script>  <script src="./lib/three/examples/js/controls/OrbitControls.js"></script>  <script src="./ParticleSystem.js"></script>  <script src="./AdditiveTerrain.js"></script>  <script>      var container;      var camera, scene, renderer, controls;        var terrain = new AdditiveTerrain();        var min = new THREE.Vector3(-900.0, 0.0, -900.0);      var max = new THREE.Vector3( 900.0, 0.0,  900.0);      var iSize = 100;      var jSize = 100;      var totalNum = 10000;      var ptSize = 20;      var texPath = "./textures/grass.jpg";      var decalPath = "./textures/leaf.png";        var terrainParams = {terrainImage: texPath, decalImage: decalPath, decalColor:0xffaa00};      terrain.initialize(min, max, iSize, jSize, totalNum, ptSize, terrainParams);        var particleSystem = new ParticleSystem();      var psParams = {blendingType : THREE.NormalBlending};      particleSystem.initialize(1000, psParams);        particleSystem.setParameters({          seedVelDir: new THREE.Vector3(0.0,-1,0),          seedVelMag: 120.0,          globalForce: new THREE.Vector3(0, -0.1, 0),          windStrength: 10,          seedSize: 60,          seedLife: 5.0,          texFile: "./textures/leaf.png",          particleColor: new THREE.Color(0Xaaaa00),          angularVel : 0.005,          alpha: 1.0      });        init();      loop();        function init() {            container = document.createElement( 'div' );          document.body.appendChild( container );            camera = new THREE.PerspectiveCamera( 60, window.innerWidth / window.innerHeight, 0.01, 10000 );          camera.setLens(18);            camera.position.y = 200;          camera.position.x = 200;          camera.position.z = 500;            scene = new THREE.Scene();            scene.add(terrain.getMesh());          scene.add(terrain.getDecalMesh());          scene.add(particleSystem.getMesh());            renderer = new THREE.WebGLRenderer( { antialias: true } );          renderer.setClearColor( 0xaaaaaa );          renderer.setSize( window.innerWidth, window.innerHeight );          container.appendChild(renderer.domElement);            controls = new THREE.OrbitControls(camera, renderer.domElement);            window.addEventListener( 'resize', onWindowResize, false );      }        function onWindowResize() {          camera.aspect = window.innerWidth / window.innerHeight;          camera.updateProjectionMatrix();            renderer.setSize( window.innerWidth, window.innerHeight );      }        var fps = 40;      var time;        function loop() {          setTimeout(loop, 1000/fps);            var now = new Date().getTime();          var dt = (now - (time||now))/1000;          time = now;            var cen = new THREE.Vector3(0,500,0);          var nor = new THREE.Vector3(0,-1,0);          var rad = 500;            particleSystem.addParticlesFromDisk(1, cen, nor, rad);          particleSystem.updateParticles(dt, terrain);            controls.update();            requestAnimationFrame(render);      }        function render() {          renderer.render( scene, camera );      }  </script>  </body>  </html> |

낙엽이 쌓이는 효과를 주기 위해서 <code5>의 27번째 줄처럼 terrainParams부분에 decalImage와 decalColor을 추가해 준다. 그리고 init함수 안에서 terrain.jetDecalMesh함수를 이용해서 지면에 쌓이는 낙엽을 장면에 추가해서 화면에서 확인할 수 있도록 하면 아래의 <그림3>과 같이 낙엽이 떨어지면서 지면에 쌓이는 모습을 볼 수 있다.

|  |
| --- |
|  |
| <그림2> |