이 게시물은 peter4k님의 블로그에 2016-01-29 오후 2:11:42에 게시되었습니다.

2. 사막의 모래바람 예제 따라 하기

계정 peter4k님의 블로그

사막에서 모래 바람이 부는 듯한 환경을 만드는 것이 이번 예제의 목표이다.

이번 예제는 다음과 같은 순서로 진행할 것이다.

1. 필요한 js파일 연결하기.
2. Three.js의 기본적인 요소들 만들기.
3. 사막 지형 만들기.
4. 만들어진 사막 지형 화면에서 확인하기.
5. 모래 바람을 표현할 파티클 시스템 설정하기.
6. 모래 바람을 사막 지형에 추가하기.

**Step 1. 필요한 js파일 연결하기**

|  |  |
| --- | --- |
| <code1> | |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | <!DOCTYPE html>  <html lang="en">  <head>      <meta charset="UTF-8">      <title>Desert Wind</title>  </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 <코드 1>의 8번째에서 11번째 줄에 나타나 있는 4가지이다.

기본 HTML문서 형식을 작성한 후 body태그 안에 script태그로 각각의 js파일을 불러오자.

**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 | <!DOCTYPE html>  <html lang="en">  <head>      <meta charset="UTF-8">      <title>Desert Wind</title>  </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;        var controls;        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.z = 400;            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 );            scene.add(terrain.getMesh());      }        function onWindowResize() {          camera.aspect = window.innerWidth / window.innerHeight;          camera.updateProjectionMatrix();            renderer.setSize( window.innerWidth, window.innerHeight );      }  </script>  </body>  </html> |

새로운 스크립트 태그를 만든 다음 자바스크립트 코드를 적어 넣는다.

전역 변수로 선언해서 사용할 요소는 <코드 2>의 13번째 줄에서 16번째 줄까지 나와 있는 것처럼 5가지이다. 이 다섯 가지의 변수들은 하나의 함수에서만 사용되는 것이 아니기 때문에 전역변수로 명시했다.

Init함수에서는 위에서 선언한 변수를 이용해서 필요한 객체를 만들고 설정을 하는 일련의 과정을 수행한다. 특히, <코드 2>의 36번째 줄과 39번째 줄에서 44번째 줄까지는 브라우저의 크기 변화에 대응하는 함수를 정의하고 연결해 놓은 것이다.

13 ~ 16 : container, camera, scene, renderer, controls를 전역변수로 선언했다.

18 ~ 34 : init함수에서 camera, scene, renderer, controls을 생성하고 초기화 한다.

36 : 브라우저의 창의 변화라는 이벤트를 처리하기 위한 함수를 연결한다.

39 ~ 44 : 브라우저의 창의 변화에 대응하는 함수를 만들었다. camera의 aspect값을 조절하고 renderer의 크기를 바꾼다.

**Step 3. 사막지형 만들기**

|  |  |
| --- | --- |
| <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 | <!DOCTYPE html>  <html lang="en">  <head>      <meta charset="UTF-8">      <title>Desert Wind</title>  </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;        var controls;        var min = new THREE.Vector3(-300, 0, -300);      var max = new THREE.Vector3( 300, 0,  300);      var iSize = 1000;      var jSize = 1000;      var totalNum = 0;      var ptSize = 0;      var texPath = "./textures/desert6.jpg";        var terrain = new AdditiveTerrain();      terrain.initialize(min, max, iSize, jSize, totalNum, ptSize, texPath);        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.z = 400;            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 );            scene.add(terrain.getMesh());      }        function onWindowResize() {          camera.aspect = window.innerWidth / window.innerHeight;          camera.updateProjectionMatrix();            renderer.setSize( window.innerWidth, window.innerHeight );      }  </script>  </body>  </html> |

<코드 3>의 18번째 줄부터 24번째 줄까지를 보면 사막 지형을 만들기 위해서 사용하는 변수들이 정의 되어있다. min과 max는 지형의 크기를 나타내고 iSize와 jSize는 지형의 텍스처를 입힐 비율을 나타낸다. totalNum, ptSize는 지면에 쌓이는 입자를 표현하기 위한 것인데 여기서는 사용하지 않기 때문에 0으로 뒀다. texPath는 지면의 텍스처를 설정하는데 사용하는 파일의 경로를 나타낸다.

<코드 3>의 26번째 줄에서 terrain이라는 변수를 사용해서 AdditiveTerrain 클래스의 객체를 만들었다. 그리고 27번째 줄에서 initialize함수를 사용해서 AdditiveTerrain클래스의 변수들을 초기화했다. 초기화를 마친 terrain객체는 장면에 추가해서 render을 할 때 보이도록 한다.

18 ~ 24 : 지면을 만들기 위한 값들을 저장하는 변수들을 정의

26 : terrain이라는 이름으로 AdditiveTerrain객체를 만들었다.

27 : AdditiveTerrain객체를 위에서 정의한 변수를 사용해서 초기화를 했다.

49 : terrain을 장면에 추가했다.

**Step 4. 만들어진 사막지형 화면에서 확인하기**

|  |  |
| --- | --- |
| <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 | <!DOCTYPE html>  <html lang="en">  <head>      <meta charset="UTF-8">      <title>Desert Wind</title>  </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;        var controls;        var min = new THREE.Vector3(-300, 0, -300);      var max = new THREE.Vector3( 300, 0,  300);      var iSize = 500;      var jSize = 500;      var totalNum = 0;      var ptSize = 0;      var texPath = "./textures/desert5.jpg";        var terrain = new AdditiveTerrain();      terrain.initialize(min, max, iSize, jSize, totalNum, ptSize, texPath);        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.z = 400;            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 );            scene.add(terrain.getMesh());      }        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> |

지금까지 만든 지면을 화면에서 확인하기 위해서는 render함수를 사용해야 한다.

기본적으로 renderer에서 제공하는 render함수를 사용하지만 여기에서 추가적으로 작업해야 할 부분이 있다. loop함수를 만들어서 주기적으로 render함수를 부르게 해야 한다는 것이다. 마우스로 시점을 조작하는 것이나 이후에 만들 모래 바람의 변화를 화면에 그때그때 반영하기 위해서는 render함수가 주기적으로 실행될 필요가 있다.

위에 대한 부분은 <코드 4>의 62번째 줄부터 74번째 줄까지에 걸쳐 나타나 있다. loop함수에서 setTimeout을 사용해서 주기적으로 loop를 부른다. 그리고 loop에서 호출하는 render함수로 화면에 장면을 그려준다.

<코드 4>를 브라우저에서 실행시키면 아래의 <그림 1>과 같은 사막지형을 볼 수 있다.

29 ~ 30 : init함수와 loop함수를 실행한다.

62 : 얼마나 자주 loop함수를 호출할 것인지를 결정. 값이 클수록 더 자주 호출한다.

65 : 1000/fps msec만큼의 시간간격을 두고 loop함수를 주기적으로 호출한다.

69 : render함수를 호출한다.

73 : renderer에서 제공하는 render함수를 이용해서 장면의 내용을 화면에 출력한다.

|  |
| --- |
| 시스템 생성 대체 텍스트: Desert Wind  C loca Ihost:63342/Particles/desertwind.htmI |
| <그림 1> |

**Step 5. 모래 바람을 표현할 파티클 시스템 설정하기**

|  |  |
| --- | --- |
| <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 | <!DOCTYPE html>  <html lang="en">  <head>      <meta charset="UTF-8">      <title>Desert Wind</title>  </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;        var controls;        var min = new THREE.Vector3(-300, 0, -300);      var max = new THREE.Vector3( 300, 0,  300);      var iSize = 500;      var jSize = 500;      var totalNum = 0;      var ptSize = 0;      var texPath = "./textures/desert5.jpg";        var terrain = new AdditiveTerrain();      terrain.initialize(min, max, iSize, jSize, totalNum, ptSize, texPath);        particleSystem = new ParticleSystem();      particleSystem.initialize(500);        particleSystem.setParameters({seedSize : 1000});      particleSystem.setParameters({seedVelMag : 450});      particleSystem.setParameters({windStrength: 5});      particleSystem.setParameters({seedLife : 40});        particleSystem.setParameters({globalForce: new THREE.Vector3(-10, -200, 0)});      particleSystem.setParameters({seedVelDir: new THREE.Vector3(-1,0.2,0)});      particleSystem.setParameters({particleColor: new THREE.Color(0xcd853f)});      particleSystem.setParameters({alpha : 0.005});      particleSystem.setParameters({seedSpread : 0.1});        particleSystem.setParameters({texFile: "./textures/smokeparticle.png"});        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.z = 400;            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 );            scene.add(terrain.getMesh());          scene.add(particleSystem.getMesh());      }        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> |

이 예제에서는 모래바람을 표현하기 위해서 연기모양의 입자를 사용하기로 했다. <코드 5>의 29번째 줄에서 43번째 줄까지에 ParticleSystem 클래스의 객체를 만들고 맴버 변수를 초기화하는 코드들이 나와있다. 특히 이번 예제에서는 'paticleParameter.js'를 사용하지 않고 setParameters함수를 이용해서 각각의 값을 결정했다. 각각의 변수의 의미는 다음 <표 1>에 나온 것과 같다. 모든 변수를 설정했다면 장면에 파티클 시스템을 추가해서 render할 수 있도록 만든다.

29 : particleSystem이라는 이름으로 ParticleSystem객체를 만든다.

30 : particleSystem의 최대 입자의 개수는 500으로 한다.

32 ~ 43 : particleSystem의 맴버변수를 설정한다.

69 : particleSystem을 장면에 추가한다.

|  |  |
| --- | --- |
| seedSize | 입자의 크기 (실수) |
| seedVelMag | 입자의 속도 (실수) |
| windStrength | 바람의 세기 (실수) |
| seedLife | 입자의 수명 (실수) |
| globalForce | 중력 벡터 (3차원 벡터) |
| seedVelDir | 입자가 생성되는 방향 (3차원 벡터) |
| particleColor | 입자의 색 (Color 객체) |
| alpha | 입자의 투명도 (0-1사이의 실수) |
| seedSpread | 입자의 퍼짐 정도 (실수) |
| texFile | 입자의 모양을 나타내는 파일의 경로 |
| <표 1> | |

**Step 6. 모래바람을 사막 지형에 추가하기**

|  |  |
| --- | --- |
| <code6> | |
| 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  113  114  115 | <!DOCTYPE html>  <html lang="en">  <head>      <meta charset="UTF-8">      <title>Desert Wind</title>  </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;        var controls;        var min = new THREE.Vector3(-300, 0, -300);      var max = new THREE.Vector3( 300, 0,  300);      var iSize = 500;      var jSize = 500;      var totalNum = 0;      var ptSize = 0;      var texPath = "./textures/desert5.jpg";        var terrain = new AdditiveTerrain();      terrain.initialize(min, max, iSize, jSize, totalNum, ptSize, texPath);        particleSystem = new ParticleSystem();      particleSystem.initialize(500);        particleSystem.setParameters({seedSize : 1000});      particleSystem.setParameters({seedVelMag : 450});      particleSystem.setParameters({windStrength: 5});      particleSystem.setParameters({seedLife : 40});        particleSystem.setParameters({globalForce: new THREE.Vector3(-10, -200, 0)});      particleSystem.setParameters({seedVelDir: new THREE.Vector3(-1,0.2,0)});      particleSystem.setParameters({particleColor: new THREE.Color(0xcd853f)});      particleSystem.setParameters({alpha : 0.005});      particleSystem.setParameters({seedSpread : 0.1});        particleSystem.setParameters({texFile: "./textures/smokeparticle.png"});        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.z = 400;            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 );            scene.add(terrain.getMesh());          scene.add(particleSystem.getMesh());      }        function onWindowResize() {         camera.aspect = window.innerWidth / window.innerHeight;         camera.updateProjectionMatrix();            renderer.setSize( window.innerWidth, window.innerHeight );      }        var fps = 40;      var time=0;        function loop() {          setTimeout(loop, 1000/fps);          var now = new Date().getTime();          var dt = (now - (time||now))/1000;          time = now;            var cen1 = new THREE.Vector3(270,0,-200);          var nor1 = new THREE.Vector3(1,0,0);          var rad1 = 80;            var cen2 = new THREE.Vector3(270,0,200);          var nor2 = new THREE.Vector3(1,0,0);          var rad2 = 100;            var cen3 = new THREE.Vector3(270,0,0);          var nor3 = new THREE.Vector3(1,0,0);          var rad3 = 80;            particleSystem.addParticlesFromDisk(2, cen1, nor1, rad1);          particleSystem.addParticlesFromDisk(2, cen2, nor2, rad2);          particleSystem.addParticlesFromDisk(2, cen3, nor3, rad3);          particleSystem.updateParticles(dt, terrain);            controls.update();            requestAnimationFrame(render);      }        function render() {          renderer.render( scene, camera );      }  </script>  </body>  </html> |

모래바람을 사막 지형에 추가하기 위해서 addParticlesFromDisk함수를 사용한다.

이 함수를 사용하면 설정한 원반에서 파티클을 생성할 수 있다.

이 예제에서는 3개의 원반을 사용한다. 사용할 원반에 대한 설정은 <코드 6>의 88번째 줄에서 98번째 줄에 걸쳐서 나와있다. 각각의 원반의 위치, 방향, 크기를 설정한다. 100, 101, 102번째 줄에서 각각의 원반을 나타내는 값들을 사용해서 3개의 디스크를 이용해서 입자들을 만든다.

<코드 6>의 80, 85, 86, 87번째 줄에 나와있는 시간관련 값들은 103번째 줄의 코드에서 사용되는 dt를 위한 값들이다. 시간에 변화에 따른 파티클 시스템의 상태를 갱신하기 위해서 시간정보가 필요하다.

<코드 6>을 브라우저에서 실행시켜보면 아래의 <그림 2>와 같은 결과화면을 볼 수 있다.

80 : 시간의 변화량을 측정하기 위해서 사용하는 변수를 선언했다.

84 ~ 86 : 시간의 변화량을 구한다.

88 ~ 90 : 1번 디스크를 만드는데 사용할 값을 선언했다.

92 ~ 94 : 2번 디스크를 만드는데 사용할 값을 선언했다.

96 ~ 98 : 3번 디스크를 만드는데 사용할 값을 선언했다.

100 ~ 102 : 위에서 선언한 값들을 이용해서 모래바람을 생성하는 디스크를 3개 만들었다.

103 : 시간에 변화에 따라서 particleSystem의 상태를 갱신한다.

|  |
| --- |
| 시스템 생성 대체 텍스트: Desert Wind  f- C localhost:63342/Particles/desertwind.html |
| <그림 2> |