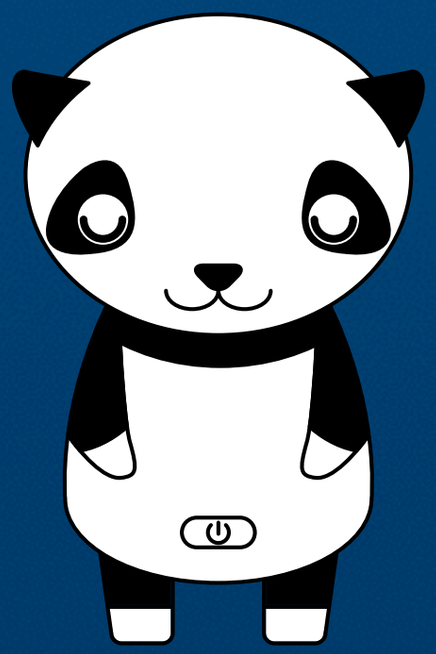
Webのグラフィックス2016



前編1:WebGL事例

事例

パフォーマンス

WebGL2

Web VR

アンケート
「〇〇についてもっと詳しく知りたい」
↓
WebGLのイベント考えます

WebGLを実務で使ったことがある？

WebGLのイメージ
これまで

- 3Dコンテンツ
- 大規模コンテンツ
- 難しい
- 重い

WebGLのイメージ
これから

- 2Dの表現でもOK
- ちょっとした表現でも使える
- 簡単に書くことも出来る
- 工夫次第で軽く

表現やコンテンツの幅を広げる選択肢の一つ

事例1
ちょっとした3D表現

ドラゴンプロジェクト：武器防具下部

http://colopl.co.jp/dragonproject/weapon/#guard

導入の経緯

- デザイン提案
- 動きのあるコンテンツで
- いろんな装備のバリエーションを見せたい
- 開発時間に余裕なし

Three.jsならいけるかも！

手順

手順

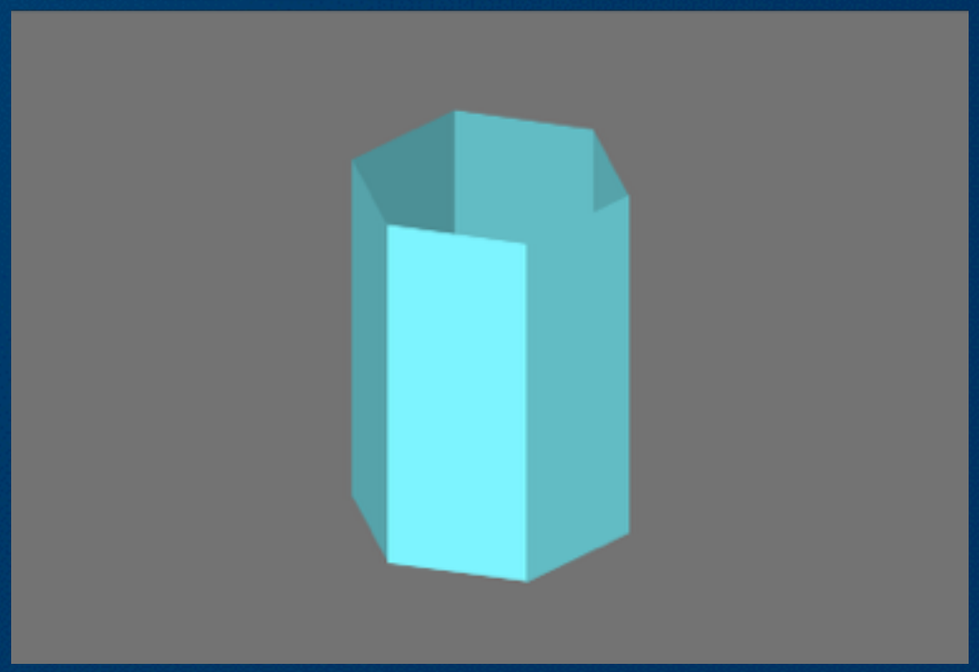
- WebGL(Three.js)を使う準備

手順

- WebGL(Three.js)を使う準備
- テクスチャにする画像を読み込む

手順

- WebGL(Three.js)を使う準備
- テクスチャにする画像を読み込む
- 空間に六角柱を準備



手順

- WebGL(Three.js)を使う準備
- テクスチャにする画像を読み込む
- 空間に六角柱を準備
- テクスチャを貼る

手順

- WebGL(Three.js)を使う準備
- テクスチャにする画像を読み込む
- 空間に六角柱を準備
- テクスチャを貼る
- 回す

```
var scene    = new THREE.Scene();
var camera   = new THREE.PerspectiveCamera( 60, window.innerWidth / window.innerHeight, 1, 1000 );
camera.position.z = 30;

var renderer = new THREE.WebGLRenderer( { antialias: true, alpha: true } );
renderer.setClearColor( 0x000000, 0 );
renderer.setSize( window.innerWidth, window.innerHeight );
```

```
var scene    = new THREE.Scene();
var camera   = new THREE.PerspectiveCamera( 60, window.innerWidth / window.innerHeight, 1, 1000 );
camera.position.z = 30;

var renderer = new THREE.WebGLRenderer( { antialias: true, alpha: true } );
renderer.setClearColor( 0x000000, 0 );
renderer.setSize( window.innerWidth, window.innerHeight );

var textureLoader = new THREE.TextureLoader();
textureLoader.load( 'texture_image2.png',
     function (texture) {
```

```
});
```

```
var scene    = new THREE.Scene();
var camera   = new THREE.PerspectiveCamera( 60, window.innerWidth / window.innerHeight, 1, 1000 );
camera.position.z = 30;

var renderer = new THREE.WebGLRenderer( { antialias: true, alpha: true } );
renderer.setClearColor( 0x000000, 0 );
renderer.setSize( window.innerWidth, window.innerHeight );

var textureLoader = new THREE.TextureLoader();
textureLoader.load( 'texture_image2.png',
     function (texture) {
          var geometry = new THREE.CylinderGeometry( 14, 14, 20, 6, 1, true );
          var material = new THREE.MeshBasicMaterial( {
               side        : THREE.DoubleSide,
               transparent : true,
               map         : texture,
               //alphaTest   : 0.3,
          } );

          var cylinder = new THREE.Mesh( geometry, material );
          cylinder.rotation.y = Math.PI / 6;
          scene.add( cylinder );

          document.body.appendChild( renderer.domElement );
```

```
});
```

```
var scene    = new THREE.Scene();
var camera   = new THREE.PerspectiveCamera( 60, window.innerWidth / window.innerHeight, 1, 1000 );
camera.position.z = 30;

var renderer = new THREE.WebGLRenderer( { antialias: true, alpha: true } );
renderer.setClearColor( 0x000000, 0 );
renderer.setSize( window.innerWidth, window.innerHeight );

var textureLoader = new THREE.TextureLoader();
textureLoader.load( 'texture_image2.png',
     function (texture) {
          var geometry = new THREE.CylinderGeometry( 14, 14, 20, 6, 1, true );
          var material = new THREE.MeshBasicMaterial( {
               side        : THREE.DoubleSide,
               transparent : true,
               map         : texture,
               //alphaTest   : 0.3,
          } );

          var cylinder = new THREE.Mesh( geometry, material );
          cylinder.rotation.y = Math.PI / 6;
          scene.add( cylinder );

          document.body.appendChild( renderer.domElement );

          var render = function (time) {
               requestAnimationFrame( render );
               TWEEN.update();
               renderer.render( scene, camera );
          };

          var flipTimer = setInterval(function(){
               var angle = cylinder.rotation.y + Math.PI / 3;
               var tween = new TWEEN.Tween(cylinder.rotation).to({y:angle},
400).easing(TWEEN.Easing.Exponential.InOut).start();
          }, 3600);

          render();
});
```

```
var scene    = new THREE.Scene();
var camera   = new THREE.PerspectiveCamera( 60, window.innerWidth / window.innerHeight, 1, 1000 );
camera.position.z = 30;

var renderer = new THREE.WebGLRenderer( { antialias: true, alpha: true } );
renderer.setClearColor( 0x000000, 0 );
renderer.setSize( window.innerWidth, window.innerHeight );

var textureLoader = new THREE.TextureLoader();
textureLoader.load( 'texture_image2.png',
     function (texture) {
          var geometry = new THREE.CylinderGeometry( 14, 14, 20, 6, 1, true );
          var material = new THREE.MeshBasicMaterial( {
               side        : THREE.DoubleSide,
               transparent : true,
               map         : texture,
               //alphaTest   : 0.3,
          } );

          var cylinder = new THREE.Mesh( geometry, material );
          cylinder.rotation.y = Math.PI / 6;
          scene.add( cylinder );

          document.body.appendChild( renderer.domElement );

          var render = function (time) {
               requestAnimationFrame( render );
               TWEEN.update();
               renderer.render( scene, camera );
          };

          var flipTimer = setInterval(function(){
               var angle = cylinder.rotation.y + Math.PI / 3;
               var tween = new TWEEN.Tween(cylinder.rotation).to({y:angle},
400).easing(TWEEN.Easing.Exponential.InOut).start();
          }, 3600);

          render();
});
```

ちょっとはまった
(ざっくり説明します)

- WebGLは描画順に上書きルール
- 透明なピクセルも上書きしてしまう
- materialにalphaTest:0.3,追加

■alphaTest:数値（0~1.0）
数値以下のalphaのピクセルを
描画しなくする設定

透明ピクセルの描画に注意

ポイント

- 結構簡単：30分くらい
- 意外と軽い

小規模なものから始めると徐々に覚えていける。

ハマり版
https://dl.dropboxusercontent.com/u/2732304/html5conf/ex/ex01_1.html

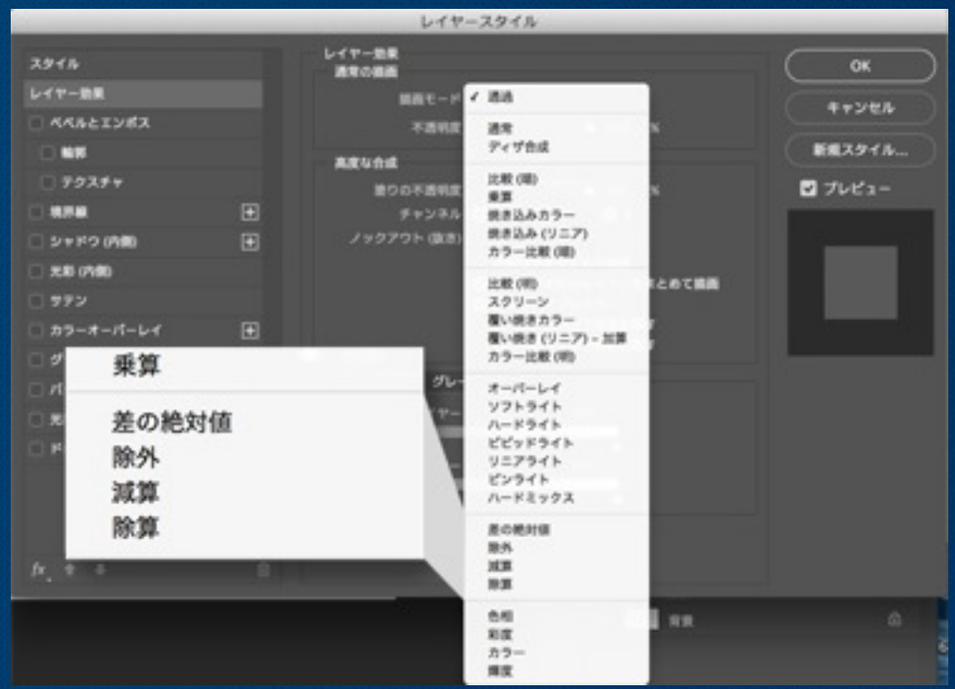
ハマり解決版
https://dl.dropboxusercontent.com/u/2732304/html5conf/ex/ex01_2.html

事例2
パーティクルでの演出

ドラゴンプロジェクト：トップ
http://colopl.co.jp/dragonproject/
白猫プロジェクト：2周年サイト
http://colopl.co.jp/shironekoproject/2nd_anniversary/sp/

導入の経緯：ドラプロ

- ✤デザイン提案
- ✤すこし複雑な動き
- ✤加算を使いたい
- ✤開発時間に余裕あり



- ✤なくても良いけどあるとより良い

Three.js+GLSLで
やってみよう！

手順：簡易版

手順：簡易版

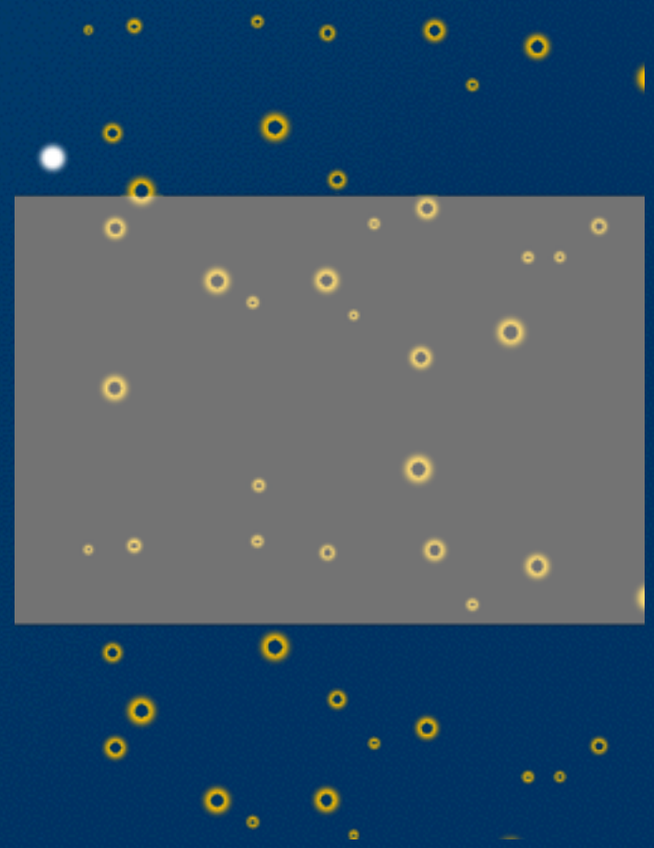
- WebGL(Three.js)を使う準備

手順：簡易版

- WebGL(Three.js)を使う準備
- テクスチャにする画像を読み込む

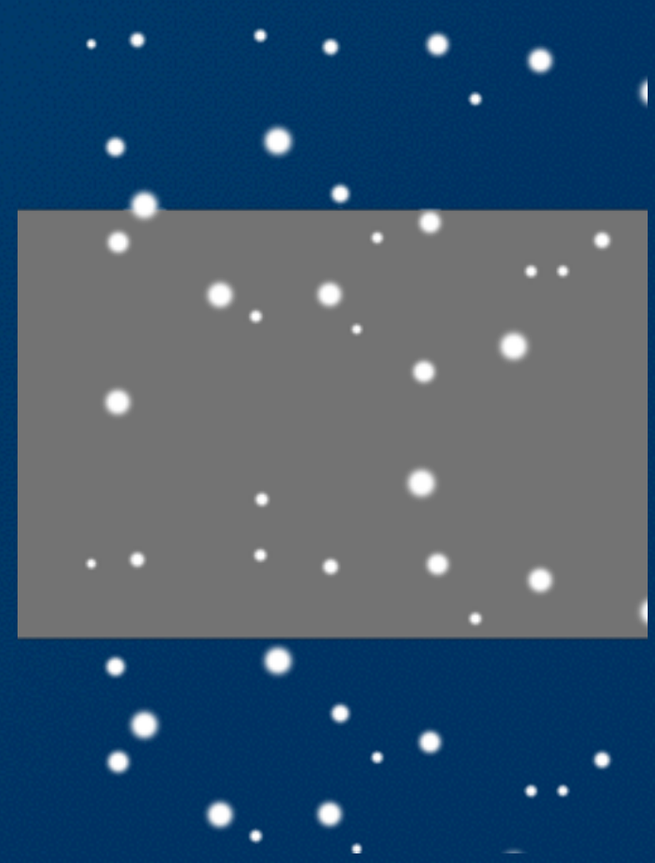
手順：簡易版

- WebGL(Three.js)を使う準備
- テクスチャにする画像を読み込む
- 空間にランダム頂点を準備



手順：簡易版

- WebGL(Three.js)を使う準備
- テクスチャにする画像を読み込む
- 空間にランダム頂点を準備
- 頂点にテクスチャを貼る（加算設定）



手順：簡易版

- WebGL(Three.js)を使う準備
- テクスチャにする画像を読み込む
- 空間にランダム頂点を準備
- 頂点にテクスチャを貼る（加算設定）
- 動かす

```
var scene    = new THREE.Scene();
var camera   = new THREE.PerspectiveCamera( 45, window.innerWidth / window.innerHeight, 1, 1000 );
camera.position.z = 100;

var renderer = new THREE.WebGLRenderer( { antialias: true, alpha: true } );
renderer.setSize( window.innerWidth, window.innerHeight );
```

```
var scene    = new THREE.Scene();
var camera   = new THREE.PerspectiveCamera( 45, window.innerWidth / window.innerHeight, 1, 1000 );
camera.position.z = 100;

var renderer = new THREE.WebGLRenderer( { antialias: true, alpha: true } );
renderer.setSize( window.innerWidth, window.innerHeight );

var textureLoader = new THREE.TextureLoader();
textureLoader.load( 'particle.png',
     function (texture) {
```

```
});
```

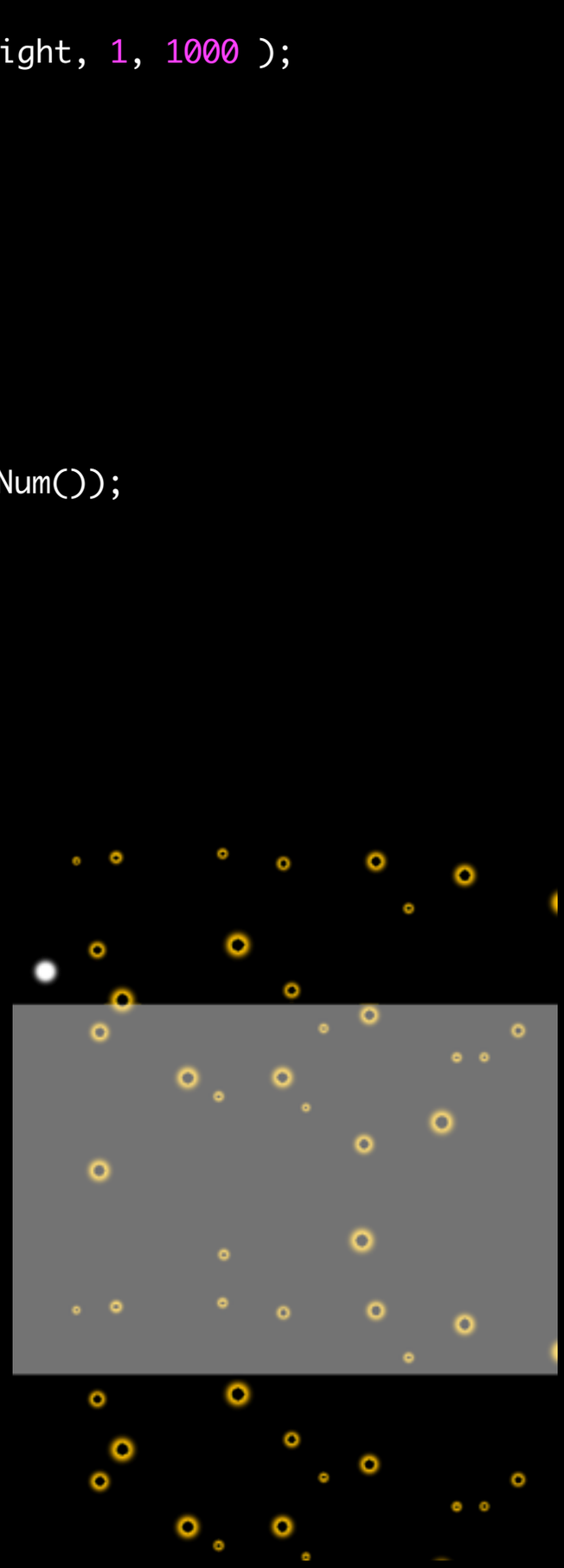
```
var scene    = new THREE.Scene();
var camera   = new THREE.PerspectiveCamera( 45, window.innerWidth / window.innerHeight, 1, 1000 );
camera.position.z = 100;

var renderer = new THREE.WebGLRenderer( { antialias: true, alpha: true } );
renderer.setSize( window.innerWidth, window.innerHeight );

var textureLoader = new THREE.TextureLoader();
textureLoader.load( 'particle.png',
     function (texture) {
          var geometry = new THREE.Geometry();
          for (var i = 0; i < count/2; i++) {
             var v1 = new THREE.Vector3(getRandomNum(), getRandomNum(), getRandomNum());
             var v2 = new THREE.Vector3(v1.x, v1.y - 100.0, v1.z);
             geometry.vertices.push(v1);
             geometry.vertices.push(v2);
          }
          var material = new THREE.PointsMaterial( {
               map         : texture,
               size        : 3,
               blending    : THREE.AdditiveBlending,
               transparent : true,
               depthTest   : false,
               //color     : 0XFF5B00,
          } );

          var particles = new THREE.Points( geometry, material );
          scene.add( particles );

          document.body.appendChild( renderer.domElement );
```



```
});
```

```
var scene    = new THREE.Scene();
var camera   = new THREE.PerspectiveCamera( 45, window.innerWidth / window.innerHeight, 1, 1000 );
camera.position.z = 100;

var renderer = new THREE.WebGLRenderer( { antialias: true, alpha: true } );
renderer.setSize( window.innerWidth, window.innerHeight );

var textureLoader = new THREE.TextureLoader();
textureLoader.load( 'particle.png',
     function (texture) {
          var geometry = new THREE.Geometry();
          for (var i = 0; i < count/2; i++) {
             var v1 = new THREE.Vector3(getRandomNum(), getRandomNum(), getRandomNum());
             var v2 = new THREE.Vector3(v1.x, v1.y - 100.0, v1.z);
             geometry.vertices.push(v1);
             geometry.vertices.push(v2);
          }
          var material = new THREE.PointsMaterial( {
               map         : texture,
               size        : 3,
               blending    : THREE.AdditiveBlending,
               transparent : true,
               depthTest   : false,
               //color     : 0XFF5B00,
          } );

          var particles = new THREE.Points( geometry, material );
          scene.add( particles );

          document.body.appendChild( renderer.domElement );

          var render = function (time) {
               requestAnimationFrame( render );
               var yPos = particles.position.y;
               particles.position.y = (yPos > 100) ? 0 : yPos + 0.2;
               renderer.render( scene, camera );
          };

          render();
});
```

```
var scene    = new THREE.Scene();
var camera   = new THREE.PerspectiveCamera( 45, window.innerWidth / window.innerHeight, 1, 1000 );
camera.position.z = 100;

var renderer = new THREE.WebGLRenderer( { antialias: true, alpha: true } );
renderer.setSize( window.innerWidth, window.innerHeight );

var textureLoader = new THREE.TextureLoader();
textureLoader.load( 'particle.png',
     function (texture) {
          var geometry = new THREE.Geometry();
          for (var i = 0; i < count/2; i++) {
             var v1 = new THREE.Vector3(getRandomNum(), getRandomNum(), getRandomNum());
             var v2 = new THREE.Vector3(v1.x, v1.y - 100.0, v1.z);
             geometry.vertices.push(v1);
             geometry.vertices.push(v2);
          }
          var material = new THREE.PointsMaterial( {
               map         : texture,
               size        : 3,
               blending    : THREE.AdditiveBlending,
               transparent : true,
               depthTest   : false,
               //color     : 0XFF5B00,
          } );

          var particles = new THREE.Points( geometry, material );
          scene.add( particles );

          document.body.appendChild( renderer.domElement );

          var render = function (time) {
               requestAnimationFrame( render );
               var yPos = particles.position.y;
               particles.position.y = (yPos > 100) ? 0 : yPos + 0.2;
               renderer.render( scene, camera );
          };

          render();
});
```

ポイント

- 加算はブレンドモードの設定のみ
- y軸 - で降る + で湧く、Z軸 + で向かってくる - で過ぎていく
- シェーダーを使えばもっと複雑に
- 描画範囲に注意

パーティクル簡易版
https://dl.dropboxusercontent.com/u/2732304/html5conf/ex/ex02.html

まとめ

- 始めるなら、ちいさなコンテンツから
- いきなり全てを理解はできない
- WebGL、そんなに難しくない（奥は深いけど
- WebGLを選択肢の一つとして加えてみよう

WebGLをつかってみましょう

WebGL事例サイト

- WebGL総本山
  https://webgl.souhonzan.org/
- ThreejsのGoogle+
  https://plus.google.com/+ThreejsOrg

Join webgl-jp on Slack
https://arcane-depths-9129.herokuapp.com/

ありがとうございました