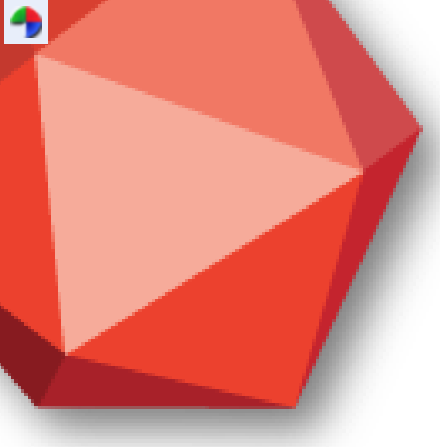


Практическое применение WebGL

Василика Климова

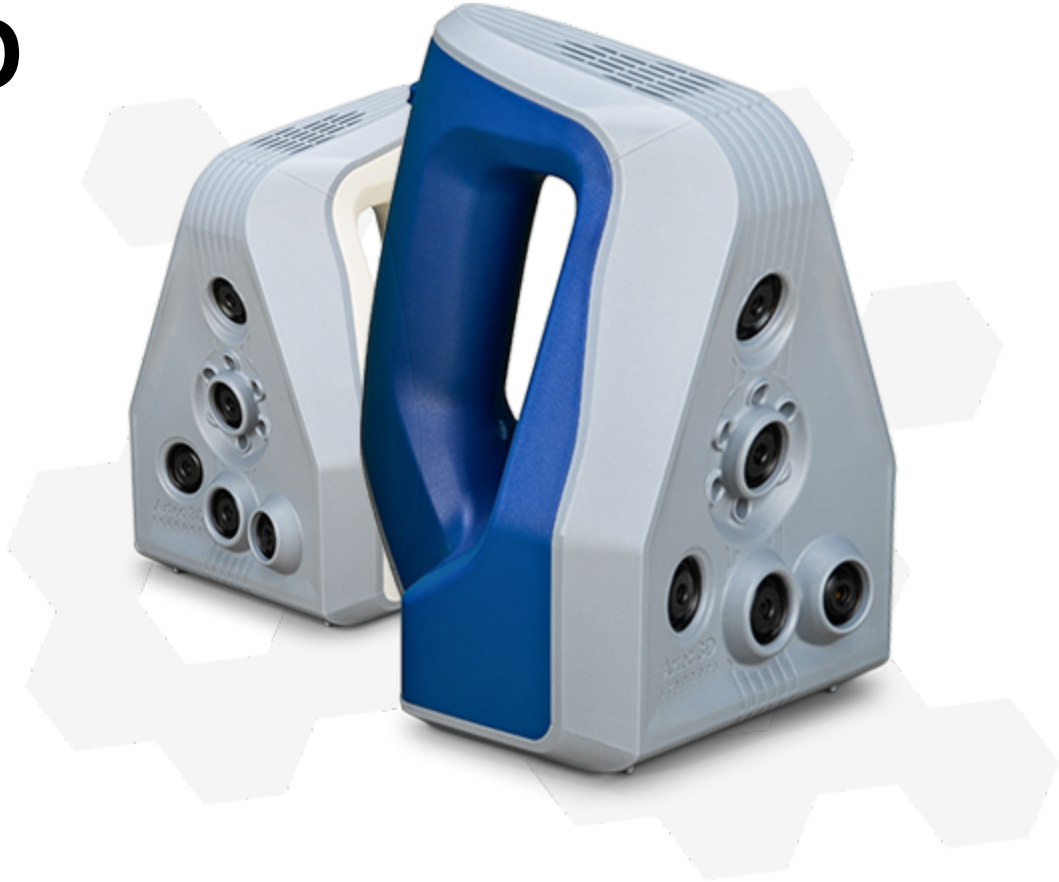
Разработчик интерфейсов

Artec 3D

@lik04ka

Artec 3D

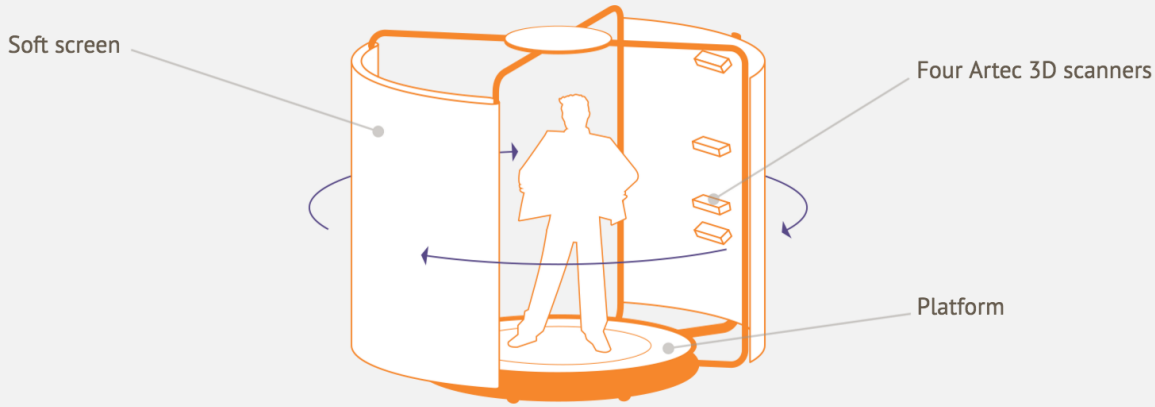
- Artec 3D
- Shapify
- Viewshape



Shapify Booth

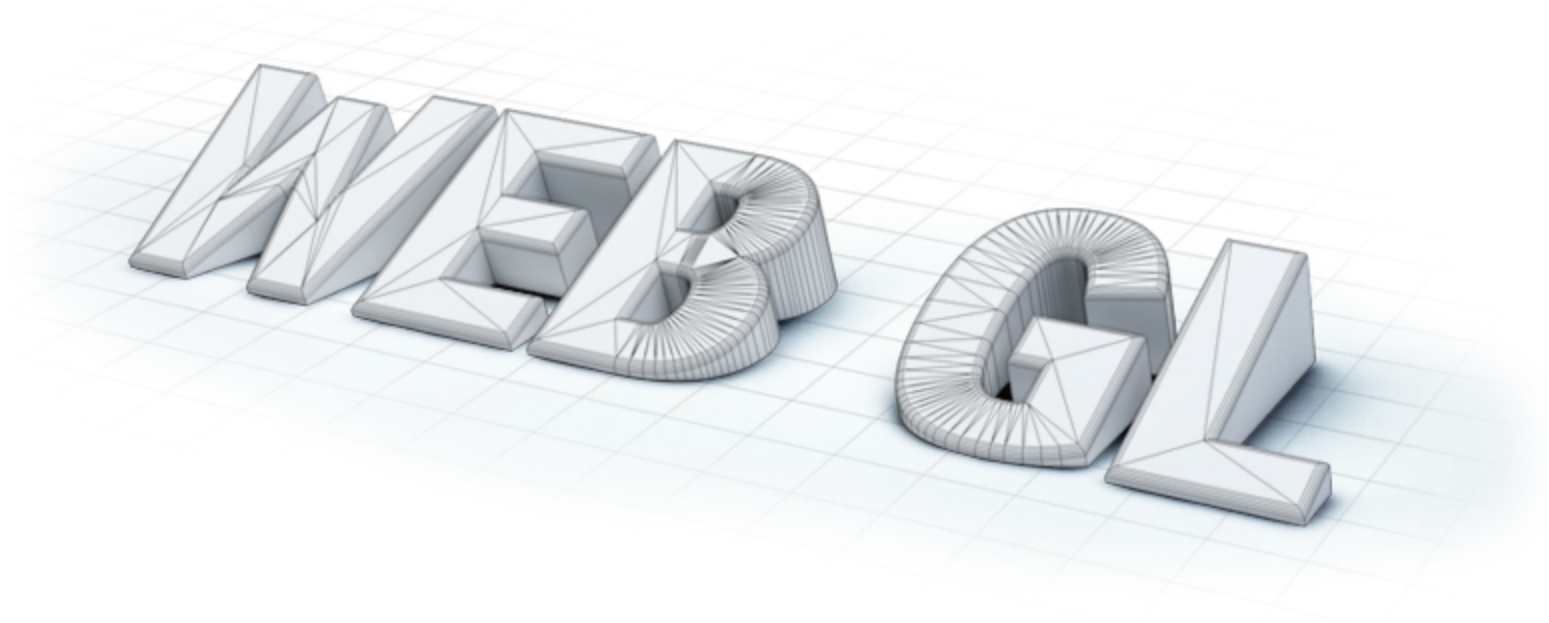
Artec Shapify Booth

Technical specs



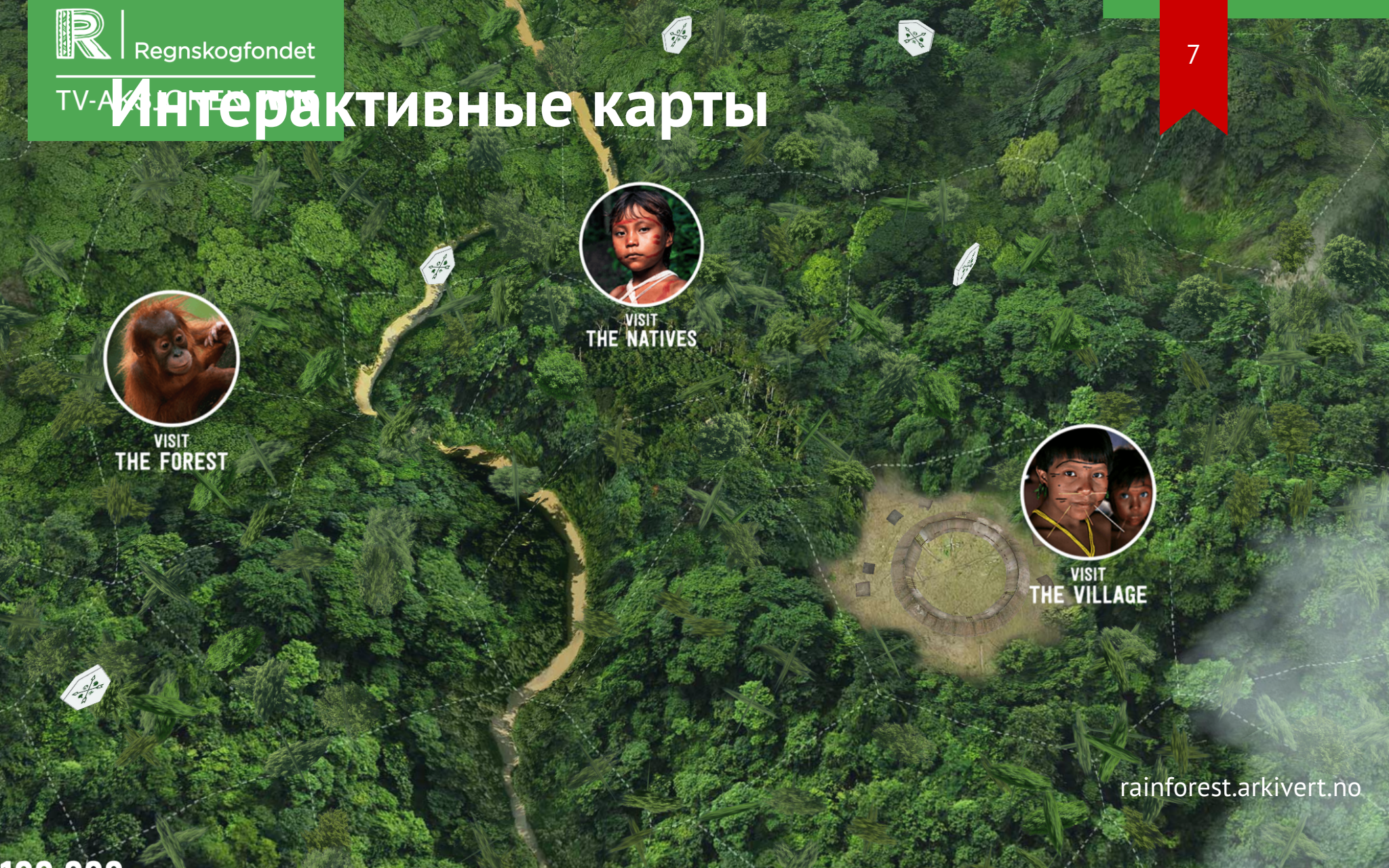
Dimensions	Scanning time	Printable model in	Polygons	Throughput
3.3x3.3x2.8 meters	12 seconds	15 minutes	400K	10 customers per hour

Содержание


WEBGL

Конструкторы

Интерактивные карты



Панорамы

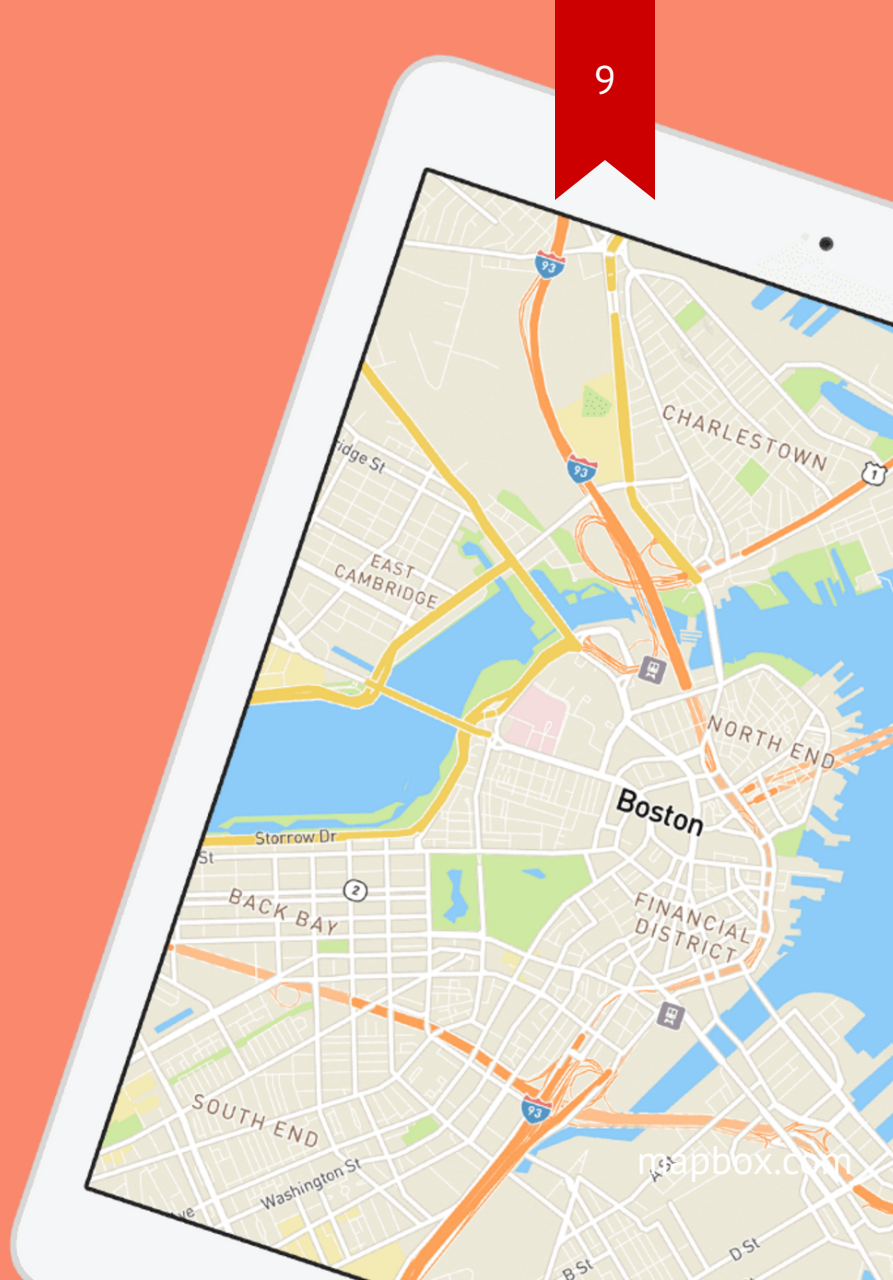
Mapbox GL JS

MAPS

Mapbox is built on vector maps, an advanced approach to mapping where data is delivered to the device and precisely **rendered in real-time**. The result is smooth, fast maps.

AVAILABLE ON: iOS Android Web

Explore features ↓



Статистика

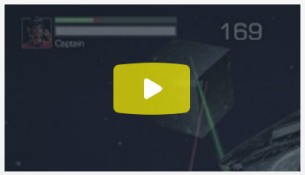

Small Arms and Ammunition — Imports & Exports
n interactive visualization of government-authorized small arms and ammunition transfers from 1992 to 2010.
RUSSIAN FED
ITALY
$0.11B
AMMUNITION
RUSSIAN FEDERATION
imported: $3
exported: $1
$9.76M
MMUNITION
$26.7M
CIVILIAN
WEAPONS
$27.7M
CIVILIAN
WEAPONS
armsglobe.chromeexperiments.com
MILITARY
CIVILIAN
AMMO
1992 93 94 95 96 97 98 99 2000 01 02 03 04 05 06 07 08 09 2010

Unity 3D

ВЕБ-ПЛАТФОРМЫ

УВЕРЕННО НАЦЕЛИВАЙТЕСЬ НА ВЕБ-ПЛАТФОРМУ

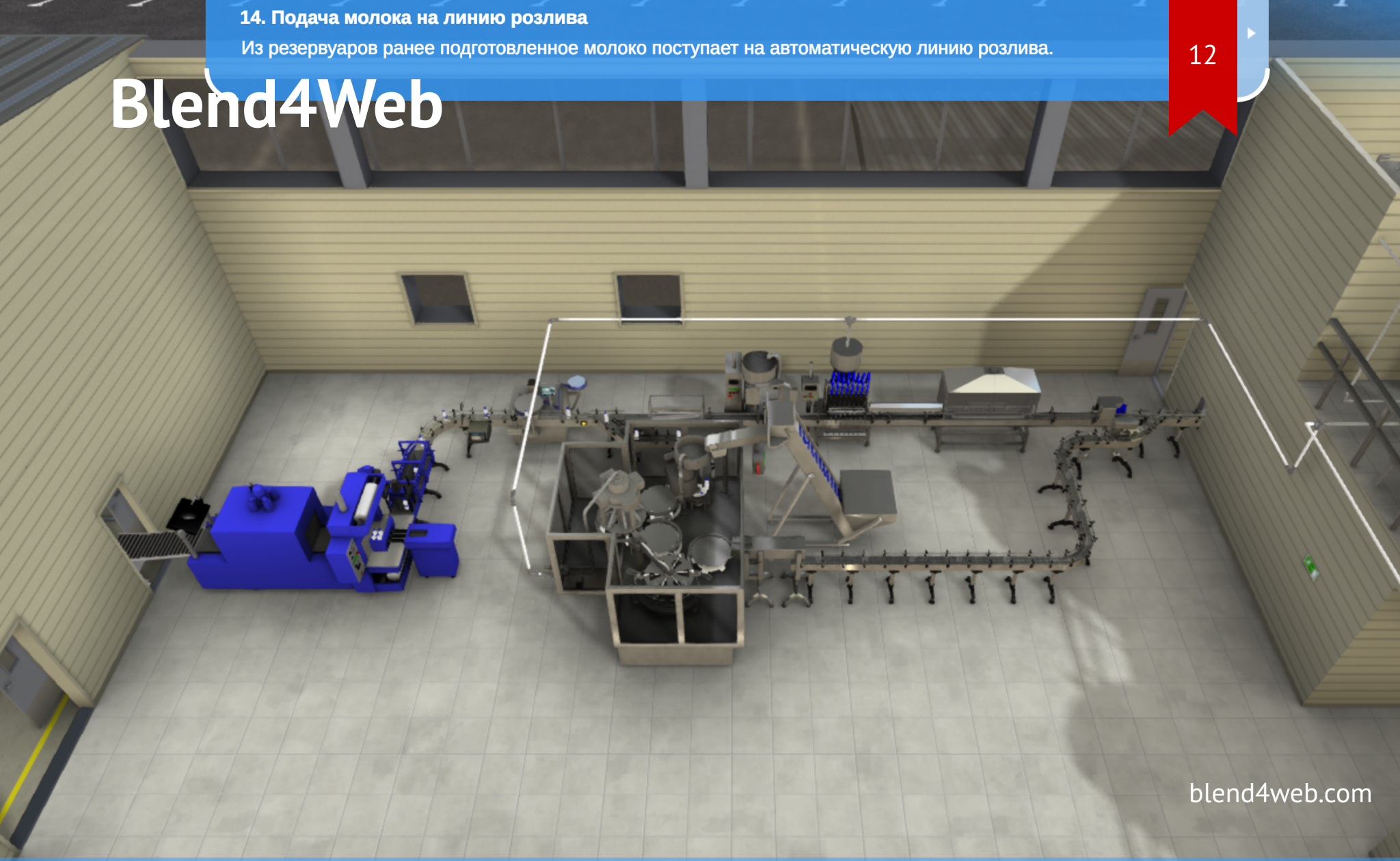
Добро пожаловать на веб без плагинов! Новая глубоко оптимизированная WebGL-сборка Unity предлагает вам нативную скорость для ваших целей. При ее помощи уже создан ряд успешных коммерческих игр.



14. Подача молока на линию розлива

Из резервуаров ранее подготовленное молоко поступает на автоматическую линию розлива.

Blend4Web



GL-React

WebGL 1.0

- HTML5 `<canvas>`
- OpenGL ES 2.0
- GLSL ES 1.1
- 2D/3D

Поддержка браузерами

WebGL - 3D Canvas graphics - OTHER

Global 56.6% + 27.44% = 84.04%

Method of generating dynamic 3D graphics using JavaScript, accelerated through hardware

Current aligned | Usage relative | Show all

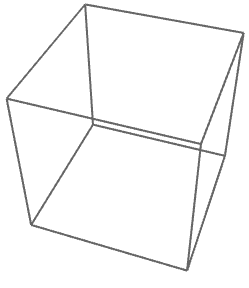
IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Chrome for Android
IEWebGL			[1] 29					
			45					
			48					
			49			8.4		
8		45	50			9.2		
[1] 11	[1] 13	46	51	9.1	37	9.3	8	50
	[1] 14	47	52	TP	38			
		48	53		39			
		49	54					

caniuse.com/webgl

Проверка поддержки WebGL

Your browser supports WebGL

You should see a spinning cube. If you do not, please
visit the support site for your browser.



get.webgl.org

WebGL

+

+ GLSL ES

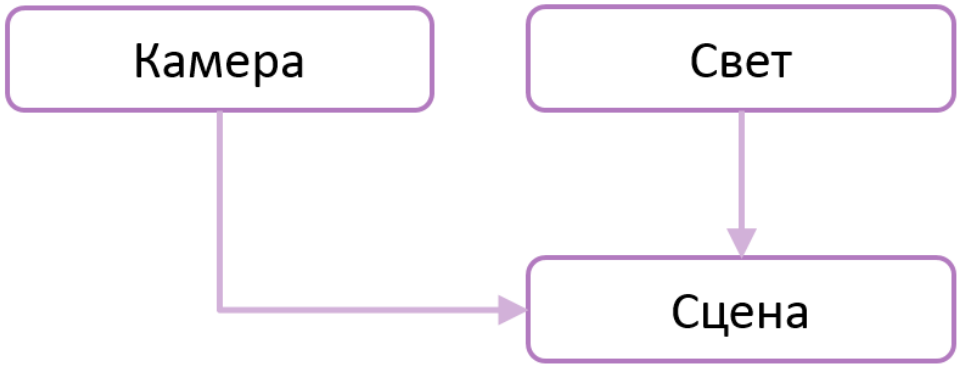
Программа на WebGL

```
1  var VSHADER_SOURCE =
2    'void main() {\n' +
3    '  gl_Position = vec4(0.0, 0.0, 0.0, 1.0);\n' +
4    '  gl_PointSize = 10.0;\n' +
5    '}\n';
6
7  var FSHADER_SOURCE =
8    'void main() {\n' +
9    '  gl_FragColor = vec4(1.0, 0.0, 0.0, 1.0);\n' +
10   '}\n';
11
12 function main() {
13   var gl = document.getElementById('webgl').getContext("webgl");
14   var vShader = gl.createShader(gl.VERTEX_SHADER);
15   gl.shaderSource(vShader, VSHADER_SOURCE);
16   gl.compileShader(vShader);
17   var fShader = gl.createShader(gl.FRAGMENT_SHADER);
18   gl.shaderSource(fShader, FSHADER_SOURCE);
19   gl.compileShader(fShader);
20   gl.program = gl.createProgram();
21   gl.attachShader(gl.program, vShader);
22   gl.attachShader(gl.program, fShader);
23   gl.linkProgram(gl.program);
24   gl.useProgram(gl.program);
25   gl.drawArrays(gl.POINTS, 0, 1);
26 }
```

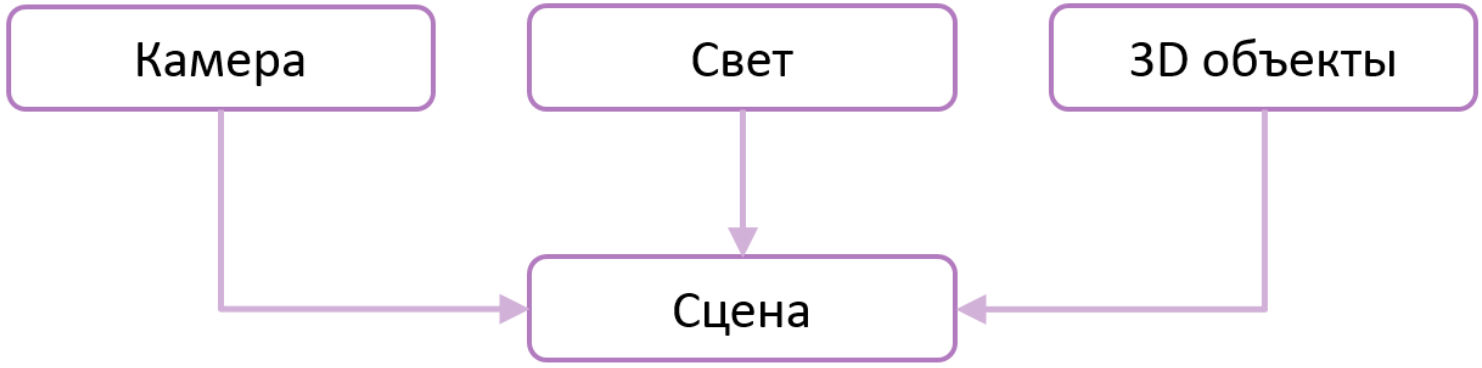
Точка

threejs.org/examples/#webgl_particles_shapes

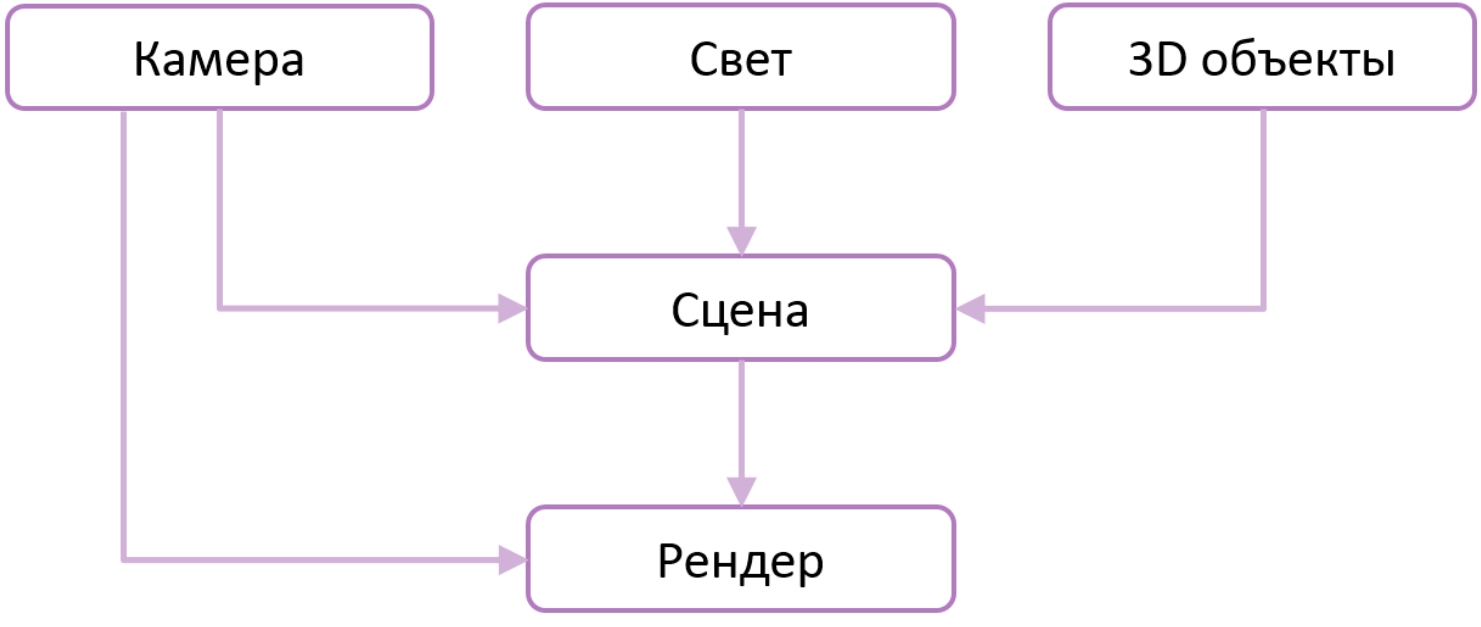
Взаимосвязь



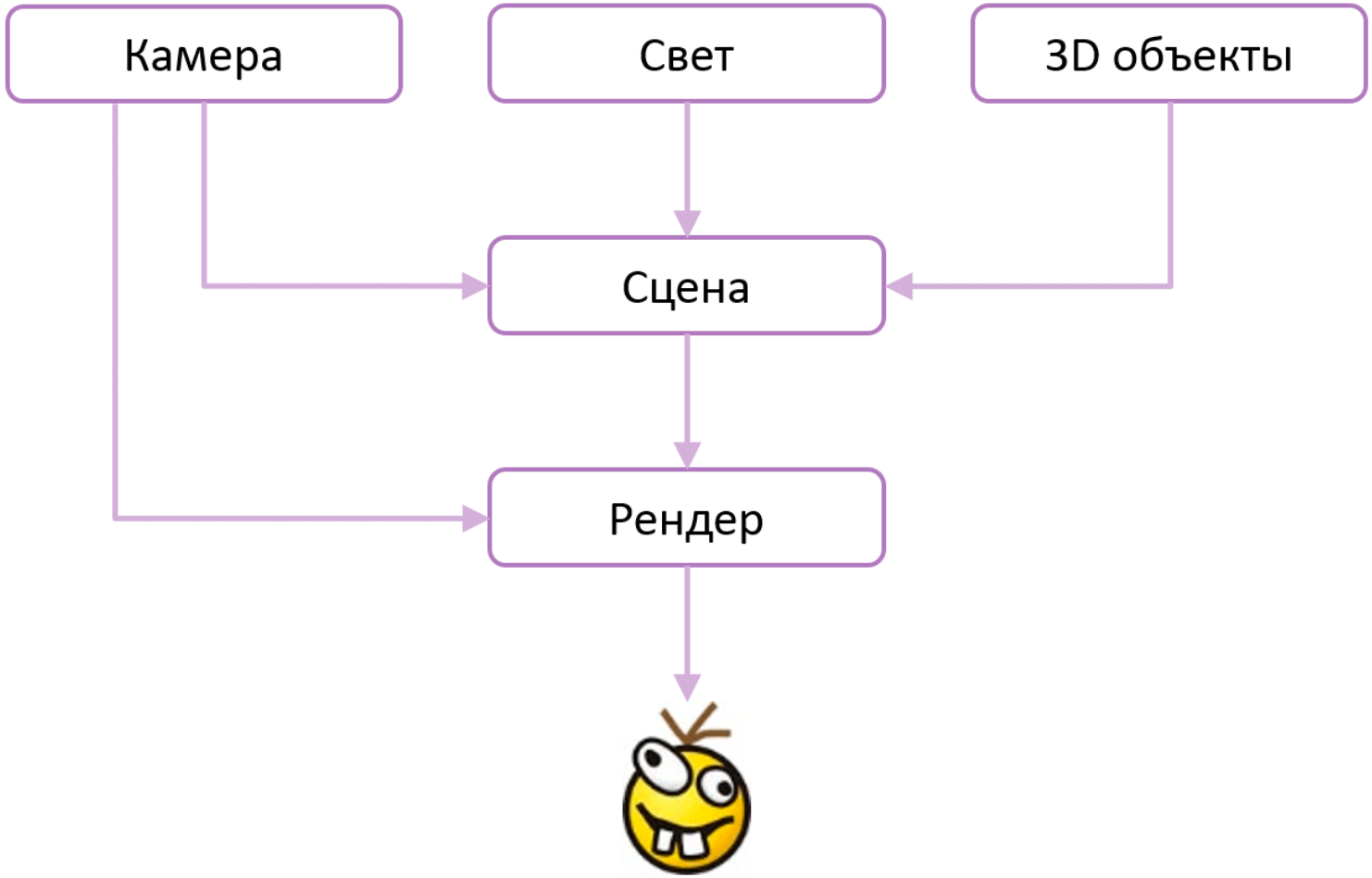
Взаимосвязь



Взаимосвязь



Взаимосвязь



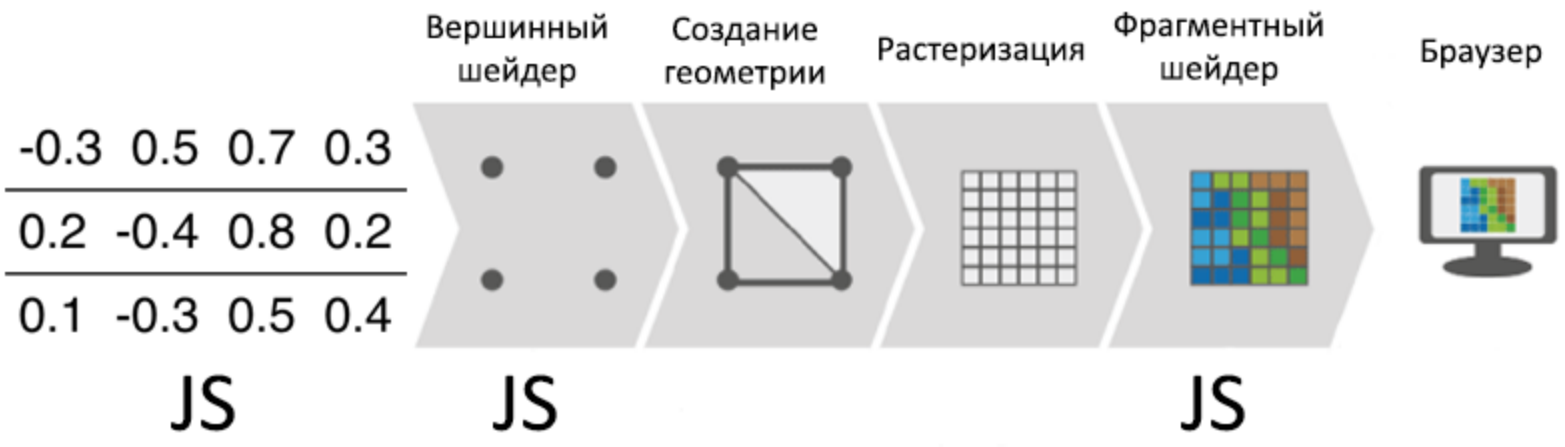

Шейдеры
MODS-PARA-MINE

GLSL

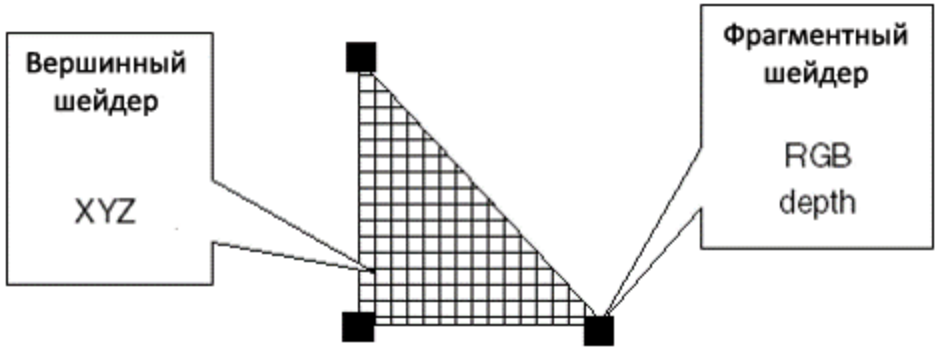
- Синтаксис C
- Типы
- Задействует GPU
- Строки
- Runtime

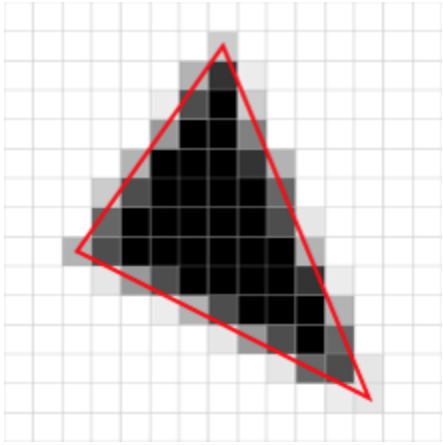
Виды шейдеров

- Вершинные шейдеры
- Фрагментные шейдеры



Растеризация



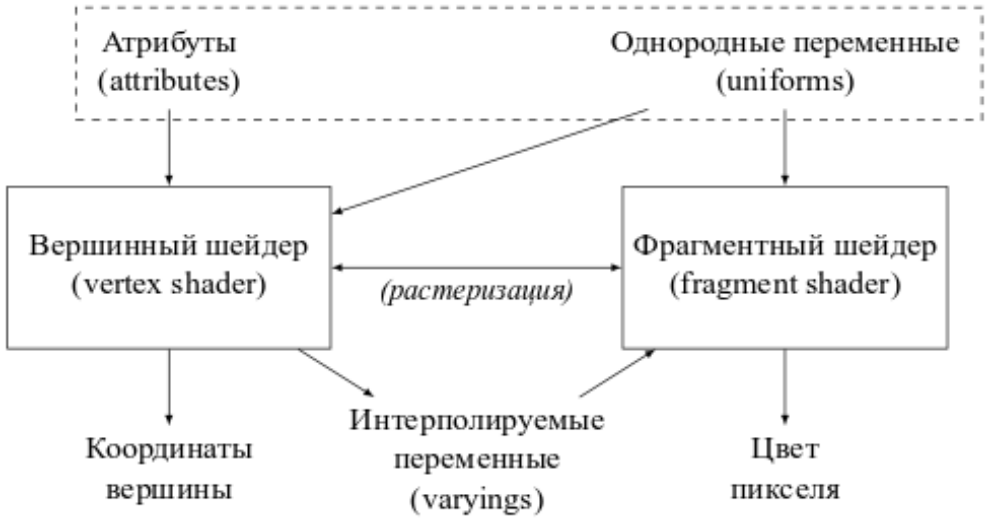


Типы данных

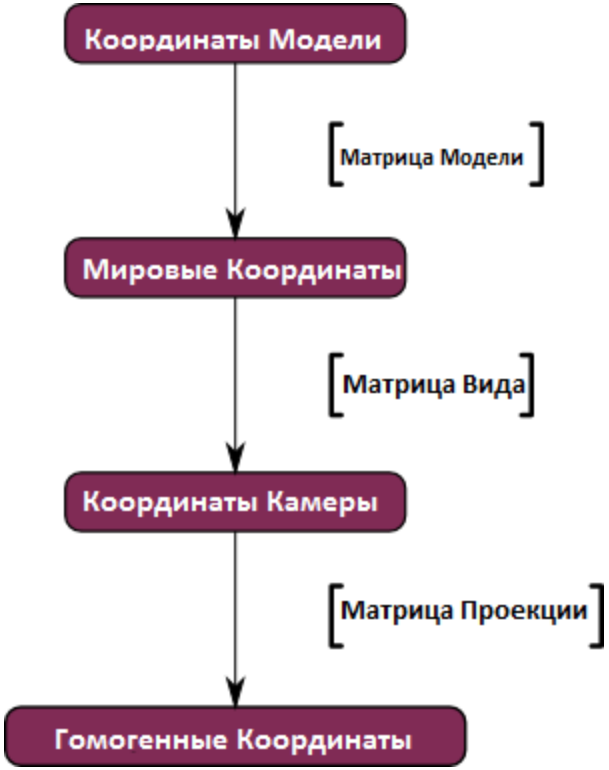
- bool
- int
- float
- vec2 / vec3 / vec4
- mat2 / mat3 / mat4
- sampler2D

Типы классификаторов

Описатели/Шейдеры	Вершинный	Фрагментный
attribute	READ	-
uniform	READ	READ
varying	READ/WRITE	READ



Матрица Модели-Вида-Проекции



Вершинный шейдер

```
01. <script type="x-shader/x-vertex" id="vshader">
02.   varying vec2 vUv;
03.   void main() {
04.     vUv = uv;
05.     vec4 mvPosition = modelViewMatrix * vec4(position, 1.0);
06.     gl_Position = projectionMatrix * mvPosition;
07.   }
08. </script>
```

Фрагментный шейдер. Текстурирование

```
01. <script type="x-shader/x-fragment" id="fshader">
02.   uniform sampler2D u_Sampler;
03.   varying vec2 vUv;
04.   void main() {
05.     gl_FragColor = texture2D(u_Sampler, vUv);
06.   }
07. </script>
```

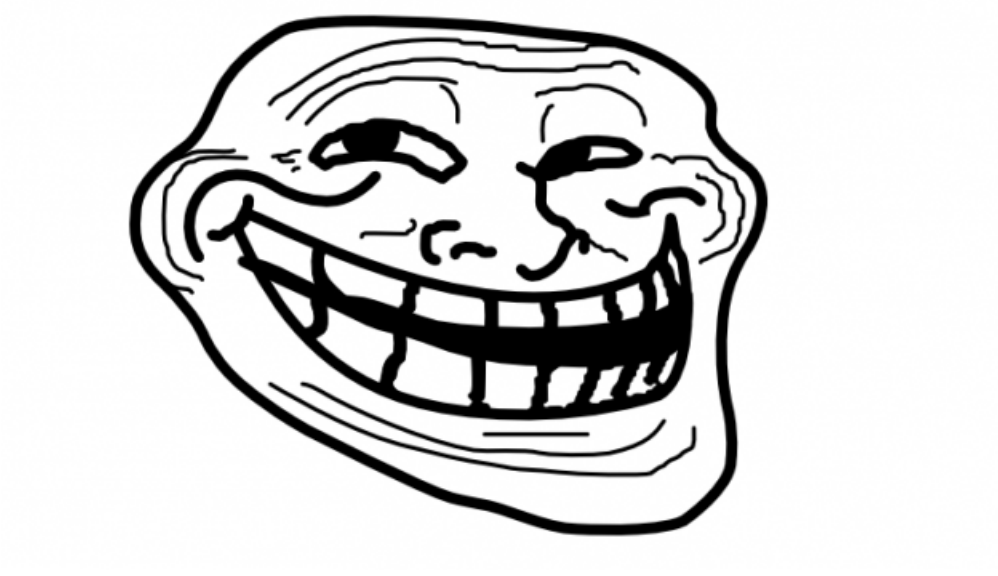
Материал из шейдера

```
01. var vShader = document.getElementById("vshader").text;
02. var fShader = document.getElementById("fshader").text;
03. object.material = new THREE.ShaderMaterial({
04.    uniforms: {
05.       u_Sampler: {type: "t", value: texture}
06.    },
07.    vertexShader: vShader,
08.    fragmentShader: fShader,
09. });
```

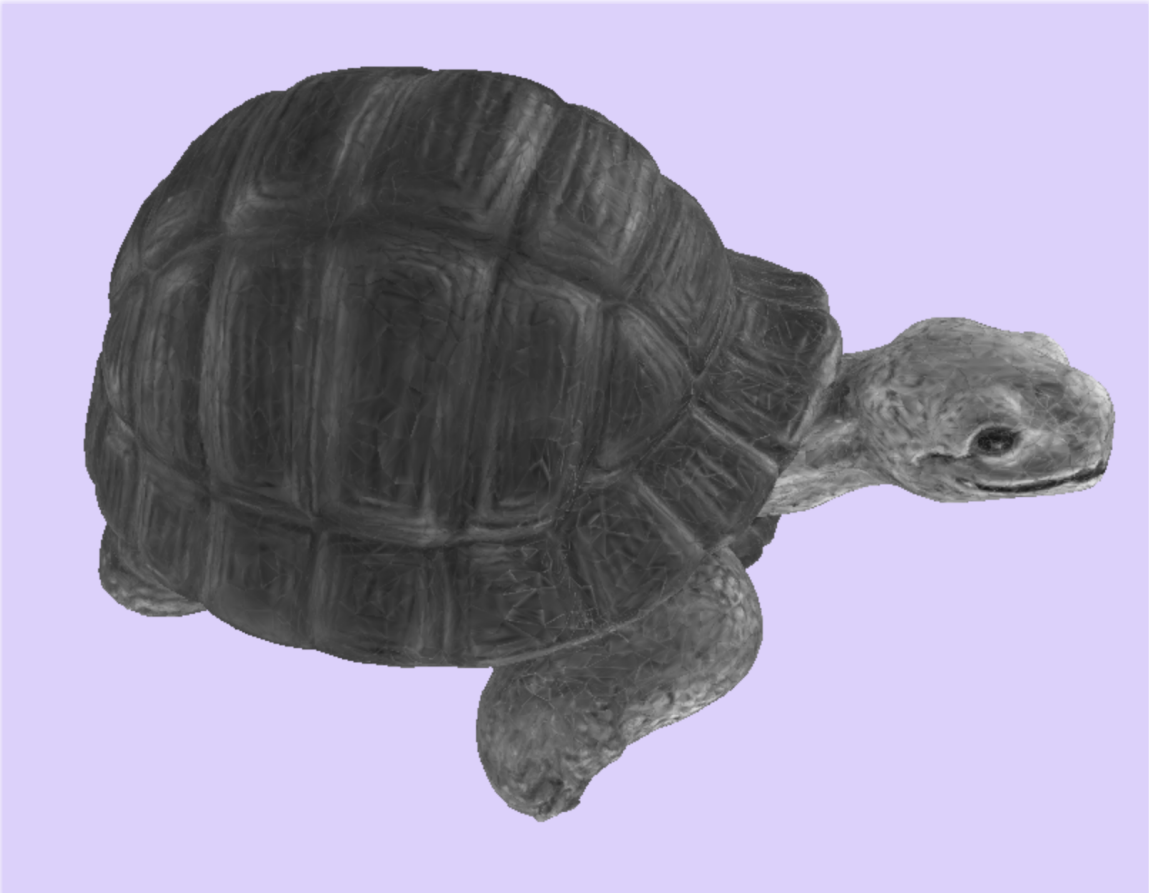
Модель с текстурой

Черно-белый фильтр

```
filter: grayscale(100%);
```



Черно-белый фильтр



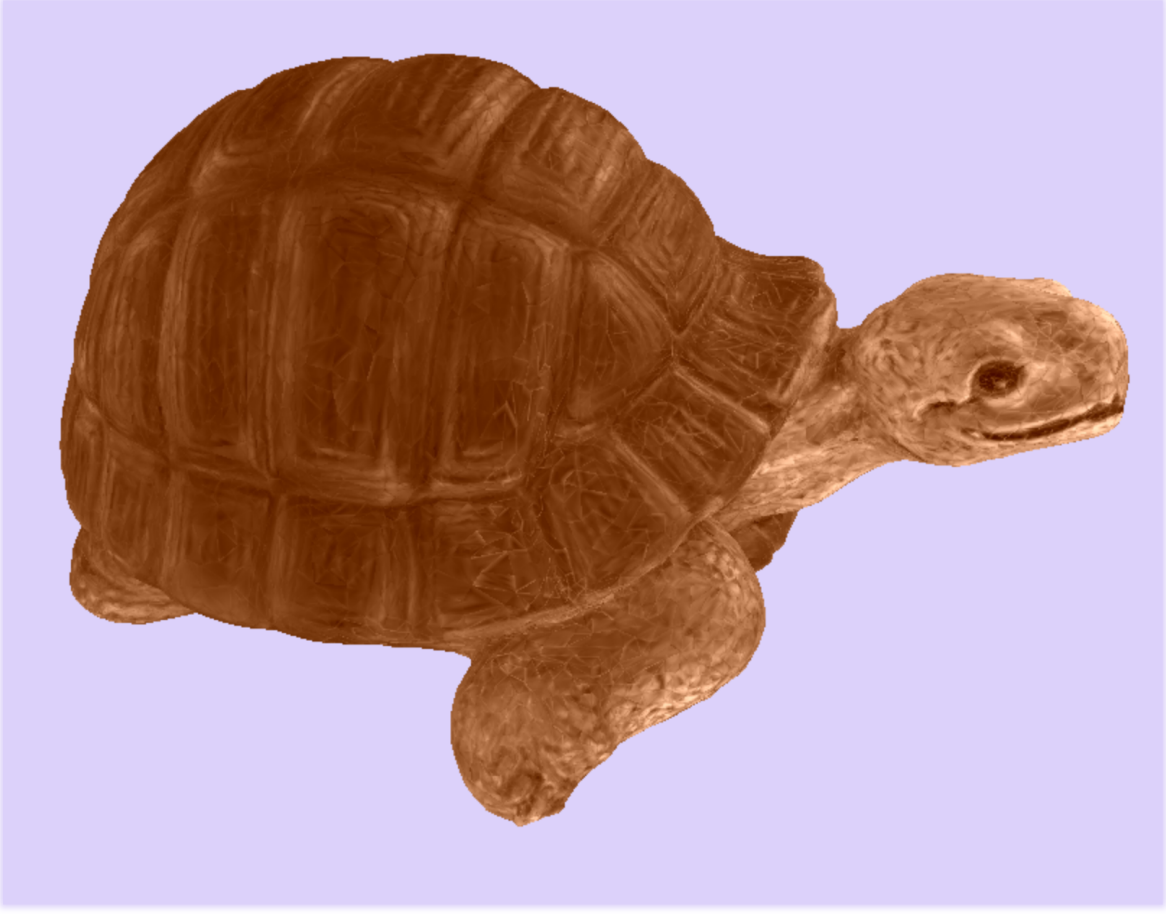
Фрагментный шейдер. ЧБ текстура

```
01. <script type="x-shader/x-fragment" id="fshader">
02. uniform sampler2D u_Sampler;
03. varying vec2 vUv;
04. void main() {
05.   vec4 color = texture2D(u_Sampler, vUv);
06.   color.rgb = vec3(color.r + color.g + color.b)/3.0;
07.   gl_FragColor = color;
08. }
09. </script>
```

Сепия

```
01. float tone = 0.299 * col.r + 0.587 * col.g + 0.114 * col.b;
02. col.r = (tone > (0.81)) ? 1.0 : tone + 0.19;
03. col.g = (tone < (0.055)) ? 0.0 : tone - 0.055;
04. col.b = (tone < (0.22)) ? 0.0 : tone - 0.22;
05. gl_FragColor = col;
```

Сепия



Отрисовка

```
01. renderer = new THREE.WebGLRenderer()
02. animate();
03. animate = function() {
04.     requestAnimationFrame(animate);
05.     renderer.render(scene, camera);
06. }
```

Cartoon шейдер. Примеры

Cartoon шейдер. Код

```
01. HueLevels[0] = 0.0;
02. HueLevels[1] = 80.0;
03. HueLevels[2] = 160.0;
04. HueLevels[3] = 240.0;
05. HueLevels[4] = 320.0;
06. HueLevels[5] = 360.0;
07. if (color == "Hue")
08.   for (int i = 0; i<HueLevCount-1; i++)
09.     if (col >= HueLevels[i] && col <= HueLevels[i+1])
10.       return HueLevels[i+1];
```

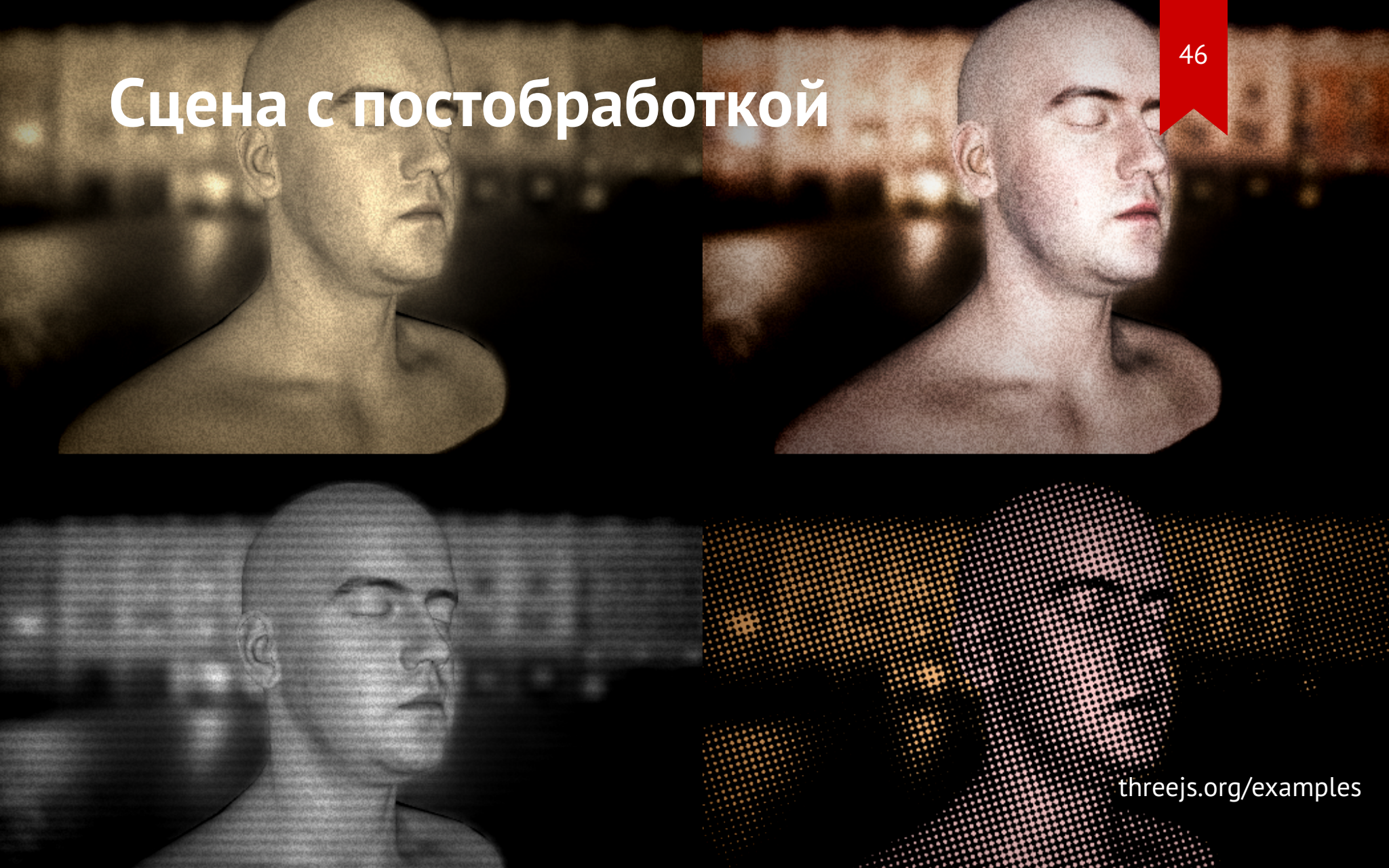
Постобработка

```
01. composer = new THREE.EffectComposer( renderer );
02. composer.addPass( new THREE.RenderPass(scene, camera) );
03. Toon = { uniforms: uniforms,
             vertexShader: vertexShader,
             fragmentShader: fragmentShader,
           };
04. effect = new THREE.ShaderPass(Toon);
05. effect.renderToScreen = true;
06. composer.addPass(effect);
```

Отображение

```
01. animate();
02. animate: function() {
03.     requestAnimationFrame(animate);
04.     composer.render();
05. }
```

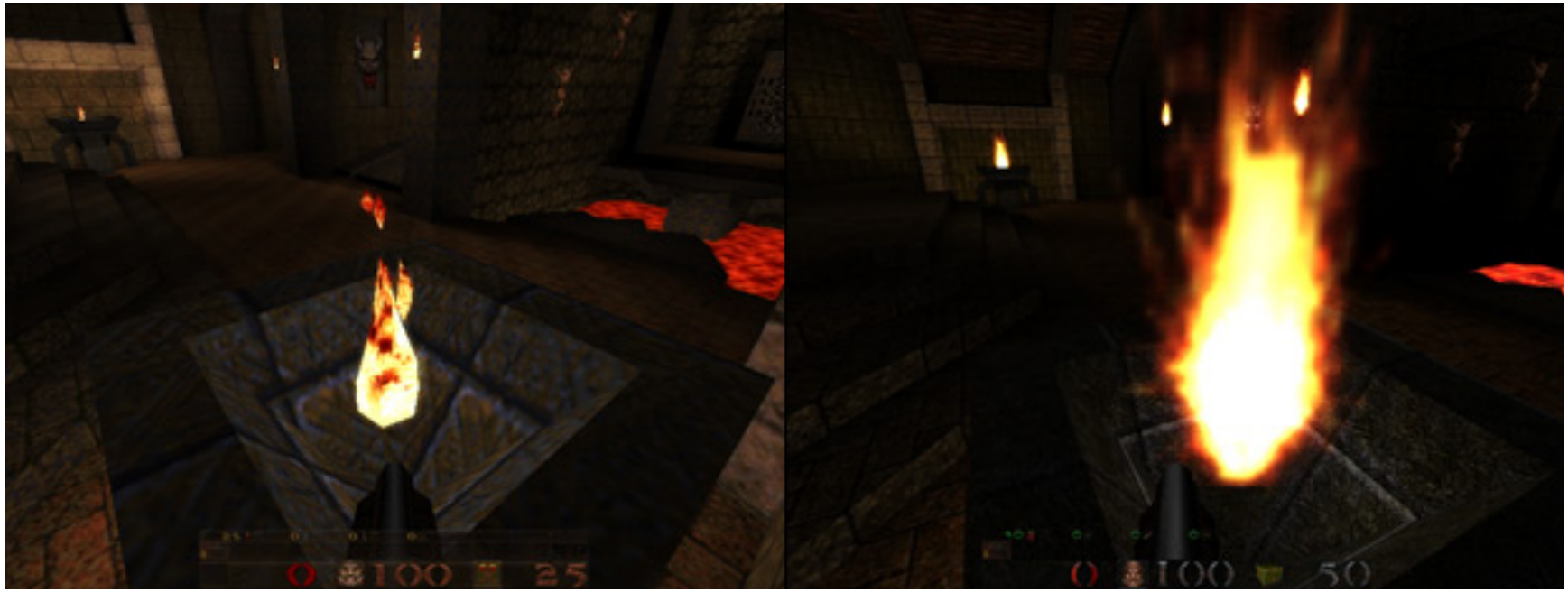
Сцена с постобработкой



threejs.org/examples

Частицы

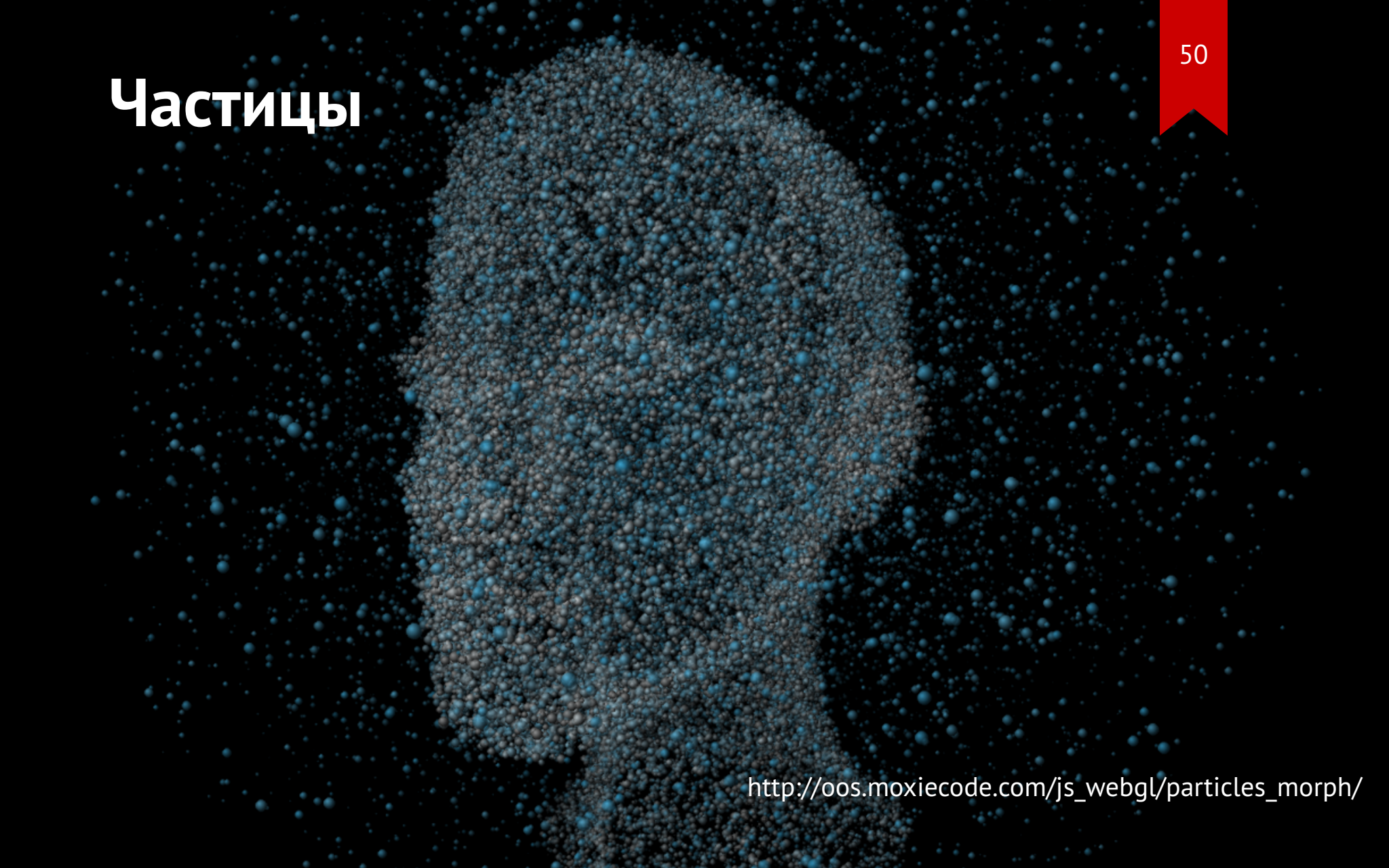
Сравнение визуализации частиц



Частицы

```
01. particleObj = new THREE.ParticleSystem(geometry, material);
02. particleObj.position.set(0, 50, 0);
03. particleObj.dynamic = true;
04. particleObj.sortParticles = true;
05. scene.add( particleObj );
```

Частицы



http://oos.moxiecode.com/js_webgl/particles_morph/

Отладка

- Three.js Inspector
- WebGL-Inspector
- Mozilla Shader Editor

Преимущества WebGL

- Производительность за счет GPU
- Отсутствие компиляции
- Кроссплатформенность
- Автоматическое управление памятью
- Открытый стандарт

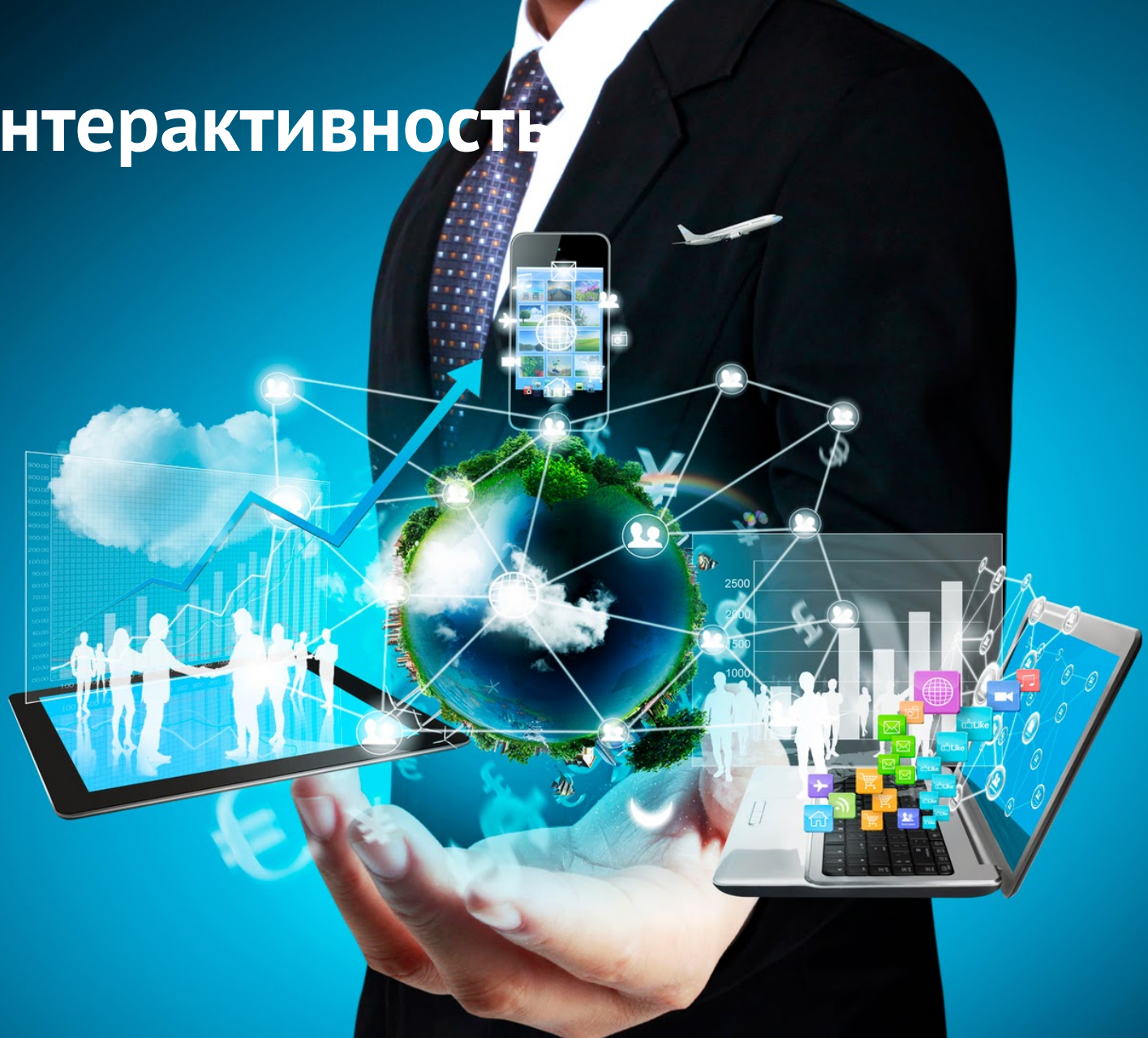
Недостатки

- Многополигональные сцены
- Требуется оптимизация
- Растровая графика

Практическое применение WebGL

- Визуализации растровой графики
- Редактирование и анализ изображений
- Фильтрация видео
- Интерактивая графика
- Игры
- Статистика в виде графиков

Интерактивность



Курс WebGL

UDACITY

Полезные ссылки

- learningwebgl.com
- webglacademy.com
- davidscottlyons.com/threejs
- Книга WebGL. Программирование трехмерной графики
- Видео уроки Никиты Северинова diductio.ru/course/2060/

Презентация

vasilika.ru

Василика Климова

likita

vasilika.klimova

lik04ka

Спасибо за внимание!