

LAB1: Introdução a Programação em Javascript e Three.js

Prof. Antonio L. Apolinário Junior
Estagiária Docente: Rafaela Alcantara

UFBA-IM-DCC-BCC 2018.1

Roteiro

- JavaScript
 - O que é
 - Variáveis, Tipos de Dados e Estruturas de Controle
 - Funções e Tratamento de Eventos
 - Objetos
 - Exemplos
- Three.js
 - O que é
 - Conceitos básicos
 - Exemplos

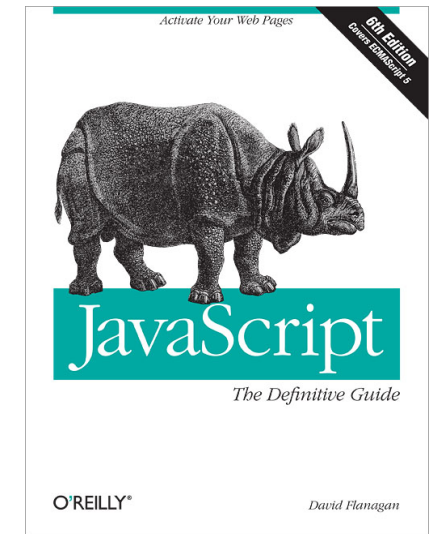
Referencias

- **JavaScript: The Definitive Guide**

6th Edition

David Flanagan

O'Reilly Media. 2011.



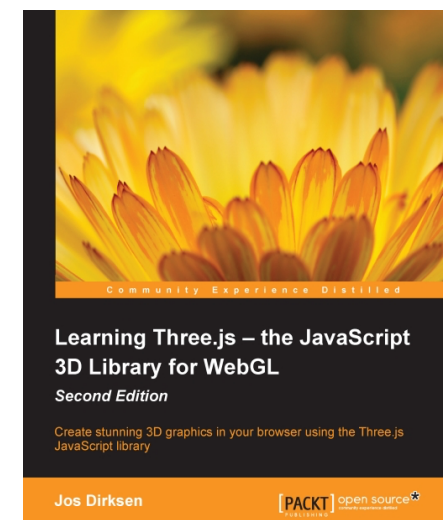
- Capítulos 1, 2 e 5

- **Learning Three.js: The JavaScript 3D Library for WebGL**

Jos Dirksen

2nd Edition.

Packt Publishing - 2015.



O que é Javascript

O que é Javascript

- Características gerais:
 - Linguagem Script
 - leve
 - interpretada
 - pelo *browser*
 - Baseada em Objetos (não orientada!)
 - Projetada para adicionar interatividade em paginas HTML
 - Código fonte pode ser escrito diretamente na página HTML
 - *Client-side dynamic content*
 - Independente de plataforma
 - Não é Java !!
 - JavaScript foi desenvolvida pela *Netscape*
 - Java foi desenvolvida pela *Sun Microsystems*

O que é Javascript

- Utilizado principalmente para:
 - Controle da interação com o usuário
 - Verificação de consistência e formatação de dados em formulários de entrada
 - Busca em pequenas bases de dados embarcadas na página
 - Salvar dados como *cookies*
- Geração dinâmica de documentos HTML

Hello World em Javascript

```
<html>
  <head>
    <title>MATA65 - Computacao Grafica</title>
    <meta http-equiv="content-type" content="text/
html; charset=UTF-8">
  </head>
  <body>
    <script type="text/javascript">
      document.write("Hello World!")
    </script>
  </body>
</html>
```

Variáveis,
Tipos de Dados e
Estruturas de Controle

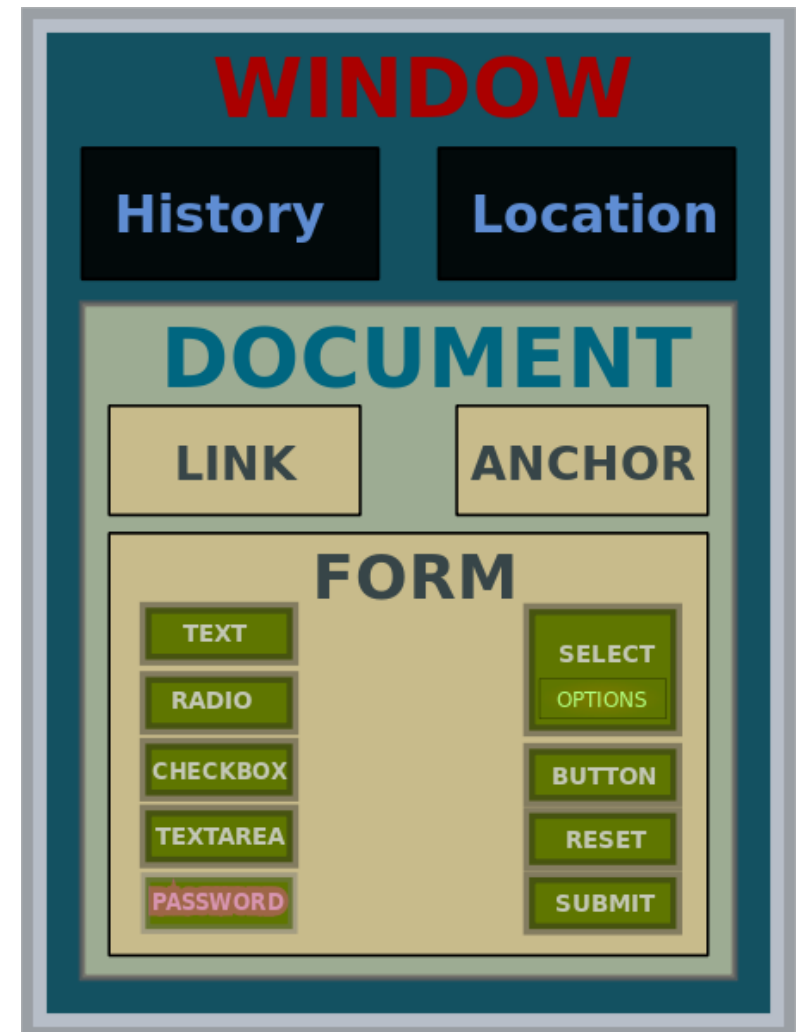
Javascript e HTML

- Acesso aos objetos HTML
 - DOM (*Document Object Model*)
- O script pode ser definido:
 - junto com o código HTML

```
<script type="text/javascript">  
    (...)   
    comandos javascript  
    (...)   
</script>
```

- em arquivo separado

```
<script type="text/javascript"  
    src="script.js"></script>
```



Características gerais

- Comentários:
 - `//` comentar uma linha ou parte dela
 - `/* */` comentar um trecho do programa
- Ponto e vírgula no final de uma linha é opcional
 - Necessário apenas quando mais de um comando estão na mesma linha

Variáveis

- Variáveis:
 - armazenam informação
 - Possuem um nome
 - Identificador alfanumérico começando por letra ou _
 - Diferencia maiúsculas e minúsculas
- Declaração:
 - Explícita:
`var i = 12;`
 - Implícita:
`i = 12; // no 'var' in declaration`

Variáveis

- Variáveis:
 - Escopo:
 - Global
 - Declarada fora de funções
 - Qualquer variável declarada implicitamente
 - Local
 - Declaração explícita dentro de uma função

Variáveis

- Tipos dinâmicos
 - Variáveis podem conter qualquer valor
- Tipos suportados:

```
var myInt = 7; // Numérico
```

```
var myBool = true; // Booleano
```

```
var myArr = new Array(); // Vetor
```

```
var myString = "abc"; // String
```

- Os valores/tipos podem mudar dinamicamente ao longo do tempo de execução

Operadores

- Aritméticos
 - $+$, $-$, $*$, $/$, ...
- Atribuição
 - $=$, $+=$, $-=$, $/=$, ...
- Relacionais
 - $==$, $!=$, $<$, $>$, ...
- Lógicos
 - $!$, $\&\&$, $||$, ...

Estruturas de Controle

- Condicionais

```
if ( boolean statement ) {  
    ... comandos ...  
}
```

```
else {  
    ... comandos ...  
}
```

```
switch ( myVar ) {  
    case 1:  ...comandos... // myVar == 1  
    case "two": ...comandos... // myVar == "two"  
    case default: ...comandos...  
}
```

Estruturas de Controle

- Repetição condicional

```
while (expressão-condicional) {  
    ... comandos ...  
}
```

```
do {  
    ... comandos ...  
} while (expressão-condicional);
```


Estruturas de Controle

- Repetição controlada

```
for(statement 1; statement 2; statement 3) {  
    ... comandos ...  
}
```

- Comandos que alteram a execução de uma repetição

break

continue

Funções e Tratamento de Eventos

Funções

- Declaração:

```
function nome(parm1, parm2) {  
    return parm1 + parm2;  
}
```

- Chamada

```
var x = nome(10, "XYZ");
```

Eventos

- Ações que ocorrem geralmente como resultado de uma interação do usuário:
 - Clique do mouse
 - Carregamento de uma página
 - Carregamento de uma imagem
 - Mouse move sobre um elemento
 - Um campo de entrada é modificado
 - Uma tecla é pressionada
 - etc...
- Event Handler
 - Função associada ao tratamento de um evento

Eventos

- Eventos de mouse
 - **onclick**
 - **ondblclick**
 - **onmousedown**
 - **onmousemove**
 - **onmouseover**
 - **onmouseout**
 - **onmouseup**
- Eventos de teclado
 - **onkeydown**
 - **onkeypress**
 - **onkeyup**
- Eventos de Objetos/
Frames
 - **onabort**
 - **onerror**
 - **onload**
 - **onresize**
 - **onscroll**
 - **onunload**

Eventos

- Para cada elemento da página que deve responder de uma forma específica a um evento, uma função de tratamento de eventos deve ser associada
 - Na definição do elemento dentro do código HTML
 - Exemplos:

```
<input type="button" onClick="doButton()>
```

```
<select onChange="doChange()">
```

```
<a onClick="doSomething()"> </a>
```

```
<form onSubmit="validate()">
```

```
<body onLoad="init()">
```

Exemplos

- Na página da disciplina:
 - <http://homes.dcc.ufba.br/~apolinario/Disciplinas/MATA65/>

Objetos

Objetos

- *Build-In Objects:*
 - **Number, Boolean, String, Date, Math,...**
- Os tipos são automaticamente aplicados a um objeto quando da atribuição de um valor.
 - **var str = "abc"; // String object**
- Podem possuir propriedades ou métodos
 - **String:**
 - **length();**
 - **toUpperCase();**
 - **substring();**

Objetos

- *Build-In Objects:*
 - **Date**
 - Não possui propriedades mas tem métodos para manipulação de datas.
 - **Math**
 - Métodos matemáticos e constantes
 - **PI**
 - **sin()**
 - **cos()**

Objetos

- *Build-In Objects:*

- **Image**

- Acesso a imagens.

```

```

...

```
<script type="text/javascript">
```

```
    function imgInfo() {
```

```
        var txt = "Imagem :" + document.getElementById("logo").src +
```

```
        "<br/> Dimensao = " +
```

```
        document.getElementById("logo").height +
```

```
        " <i>x</i> " +
```

```
        document.getElementById("logo").width;
```

```
        document.getElementById("output").innerHTML = txt;
```

```
    }
```

```
</script>
```

Objetos

- Objetos definidos pelo programador:
 - Criar um construtor;
 - Utilizar a função **new** para criar uma variável desse tipo.

Objetos

- Exemplo:

```
var objPto = function(x,y,cor) {  
    this.x    = x;  
    this.y    = y;  
    this.cor  = cor;  
    this.draw = function(ctx) {  
        ctx.fillStyle = this.cor;  
        ctx.fillRect(this.x-1, this.y-1, 2, 2);  
    }  
}  
  
var pto = new objPto(10, 10, "#FF0000");
```

Objetos

- Exemplo:

```
var objPto = function(x,y,cor) {  
    this.x    = x;  
    this.y    = y;  
    this.cor   = cor;  
};  
  
var pto = new objPto(x0, y0, "red");  
objPto.prototype.draw = function(ctx) {  
    ctx.fillStyle = this.cor;  
    ctx.fillRect(this.x - 1, this.y - 1, 2, 2);  
}
```

Objetos

- Exemplo:

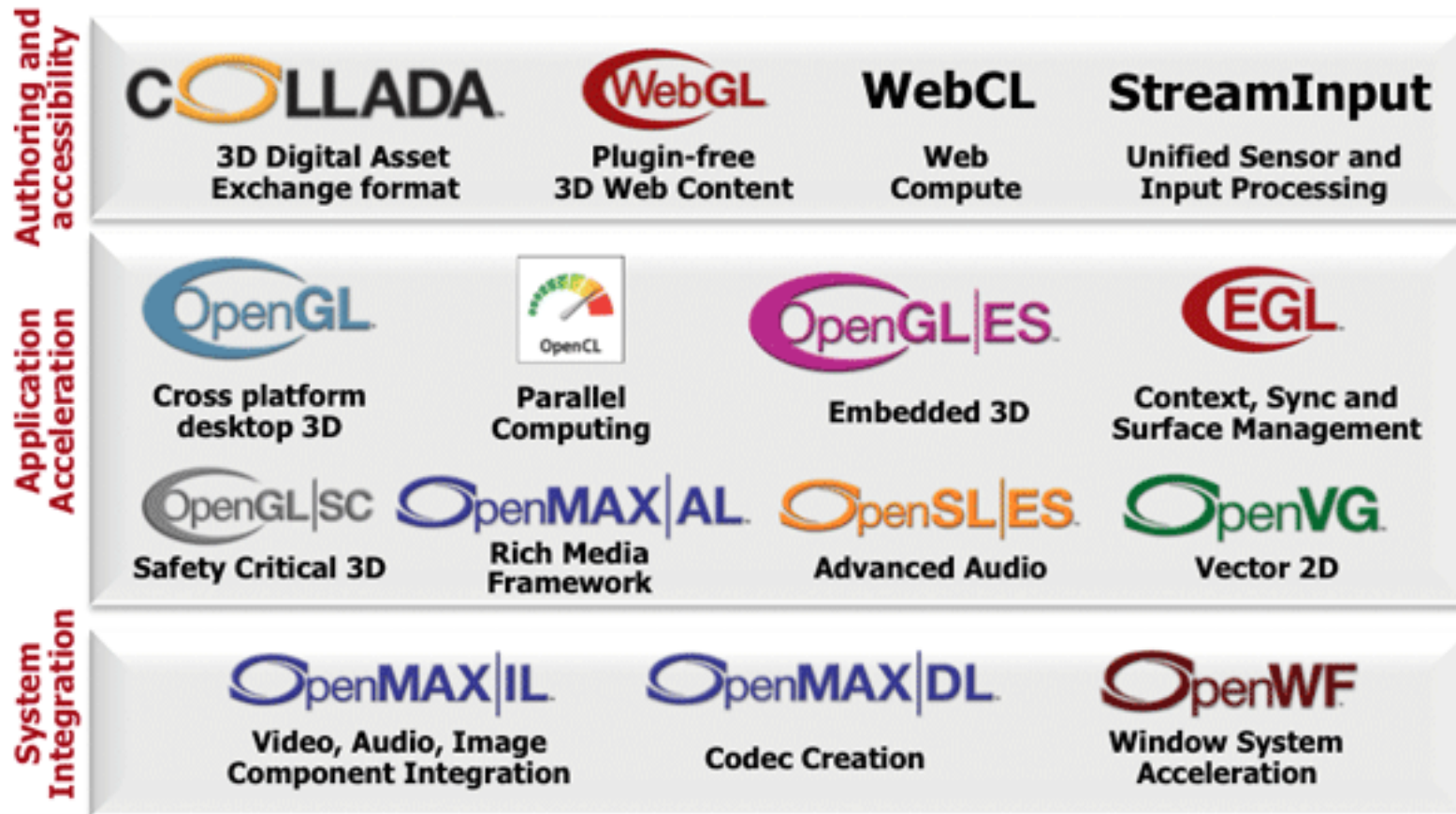
```
objPto.prototype.draw = function(ctx) {  
    ctx.fillStyle = this.cor;  
    ctx.fillRect(this.x - 1, this.y - 1, 2, 2);  
}
```

```
var objPto = function(x,y,cor) {  
    this.x      = x;  
    this.y      = y;  
    this.cor    = cor;  
    this.draw = function(ctx) {  
        ctx.fillStyle = this.cor;  
        ctx.fillRect(this.x-1, this.y-1, 2, 2);  
    }  
};
```

WebGL

O que é WebGL

- Parte do ecossistema baseado em **OpenGL** (*Open Graphics Library*)



O que é WebGL

- WebGL = ***OpenGL ES 2.0 for the Web***



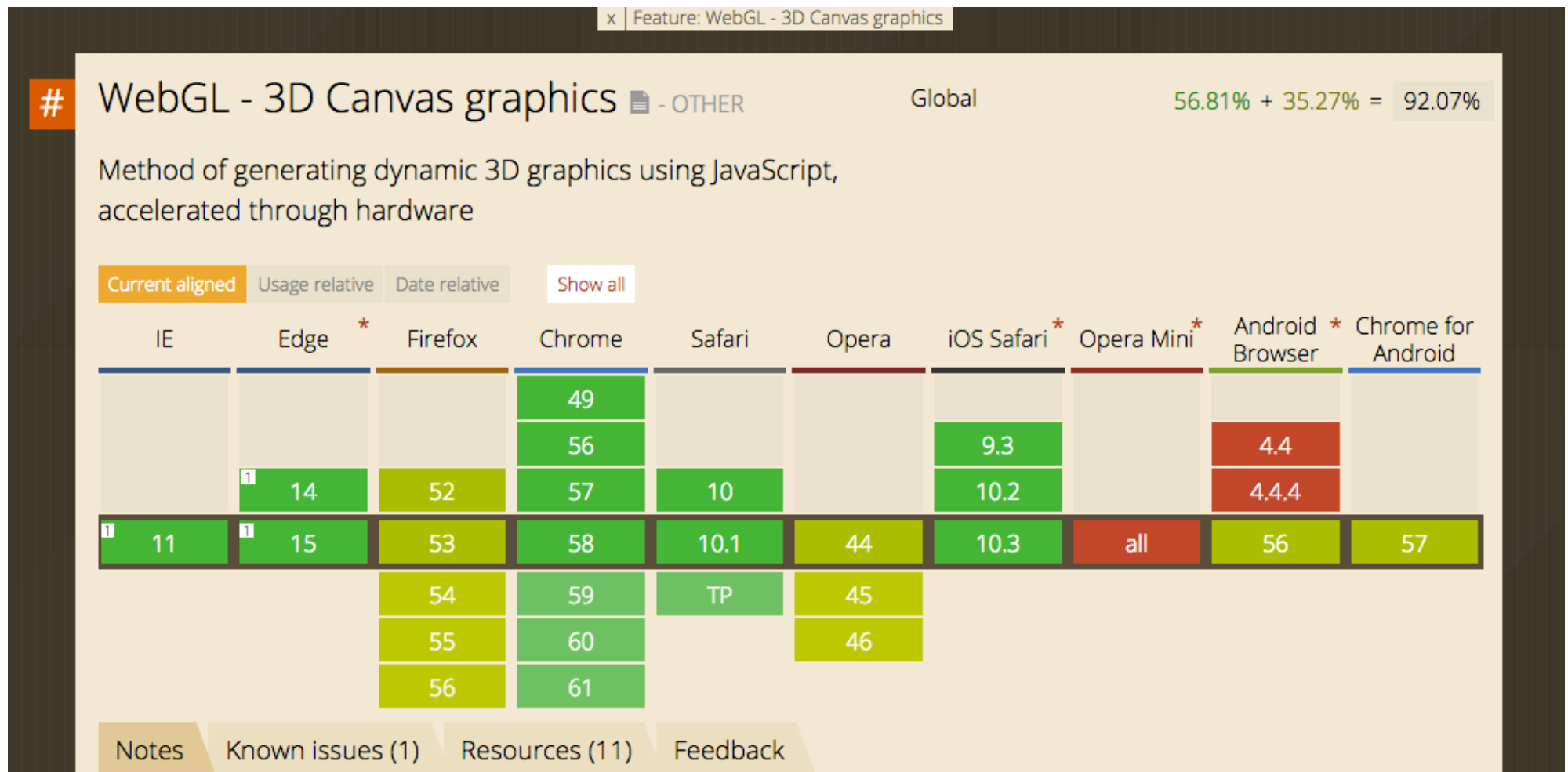
O que é WebGL

- WebGL:
 - versão “leve” da OpenGL
 - *Binding* com JavaScript
 - Suportado pelo objeto *canvas* do HTML5



O que é WebGL

- Suportado por quase todos os navegadores mais atuais:
 - <http://caniuse.com/#feat=webgl>



O que é WebGL

- Como verificar?
 - <http://get.webgl.org>

Your browser supports WebGL

However, it indicates that support is experimental; you might see issues with some content.

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



O que é WebGL

- Documentação, especificação, etc..
- <http://www.khronos.org/webgl/>

The screenshot shows the Khronos Group website. At the top, there's a navigation bar with links for English, 中文, 日本語, and 한국어. Below this is the Khronos Group logo and a tagline "CONNECTING SOFTWARE TO SILICON". A secondary navigation bar includes links for Developers, Conformance, Membership, News, Events, and Forums. A table lists various APIs: Vulkan, OpenCL, OpenGL, OpenGL ES, WebGL, WebCL, COLLADA, glTF, EGL, OpenGL ES, OpenMAX, SPIR, SYCL, StreamInput, OpenVX, OpenKCam, OpenVG, and Other. The main content area is titled "OpenGL ES 2.0 for the Web" and features the WebGL logo. It describes WebGL as a cross-platform, royalty-free web standard for a low-level 3D graphics API based on OpenGL ES 2.0. A list of links is provided at the bottom, including the WebGL 1.0 Specification, Public Wiki, Public Mailing List, Reference Card, Google Groups, StackOverflow discussions, bug filing instructions, and a Security white paper. On the right side, there are two promotional banners: "KHRONOS REFERENCE CARDS" and "IWOCCL INTERNATIONAL WORKSHOP ON OPENCL 12-13 MAY 2015 Stanford Univ. USA". At the bottom right, there's a banner for the "embedded VISION SUMMIT May 12, 2015".

English 中文 日本語 한국어 Login Members Adopters Implementers Search Khronos.org...

KHRONOS
GROUP
CONNECTING SOFTWARE TO SILICON

Developers Conformance Membership News Events Forums

Vulkan	OpenCL	OpenGL	OpenGL ES	WebGL	WebCL	COLLADA	glTF	EGL
OpenSL ES	OpenMAX	SPIR	SYCL	StreamInput	OpenVX	OpenKCam	OpenVG	Other

Home > WebGL

Overview Wiki Mailing List Resources Google Groups StackOverflow Github

WebGL

OpenGL ES 2.0 for the Web

WebGL is a cross-platform, royalty-free web standard for a low-level 3D graphics API based on OpenGL ES 2.0, exposed through the HTML5 Canvas element as Document Object Model interfaces. Developers familiar with OpenGL ES 2.0 will recognize WebGL as a Shader-based API using GLSL, with constructs that are semantically similar to those of the underlying OpenGL ES 2.0 API. It stays very close to the OpenGL ES 2.0 specification, with some concessions made for what developers expect out of memory-managed languages such as JavaScript.

WebGL brings plugin-free 3D to the web, implemented right into the browser. Major browser vendors Apple (Safari), Google (Chrome), Mozilla (Firefox), and Opera (Opera) are members of the WebGL Working Group.

- [WebGL 1.0 Specification](#)
- [WebGL Public Wiki](#)
- [WebGL Public Mailing List \(spec discussion\)](#) and [Public Mailing List Archives](#)
- [WebGL Reference Card](#)
- [Google Groups](#) and [StackOverflow](#) discussions on developing with WebGL
- [Filing bugs about the WebGL spec or conformance tests](#)
- [WebGL Security white paper](#)

KHRONOS
REFERENCE CARDS

IWOCCL
INTERNATIONAL
WORKSHOP
ON OPENCL
12-13 MAY 2015
Stanford Univ. USA

**embedded
VISION
SUMMIT**
May 12, 2015

Three.js

Three.js

- Framework para programação WebGL
- <http://www.threejs.org>

three.js ^{r85}

featured projects

[more projects](#)

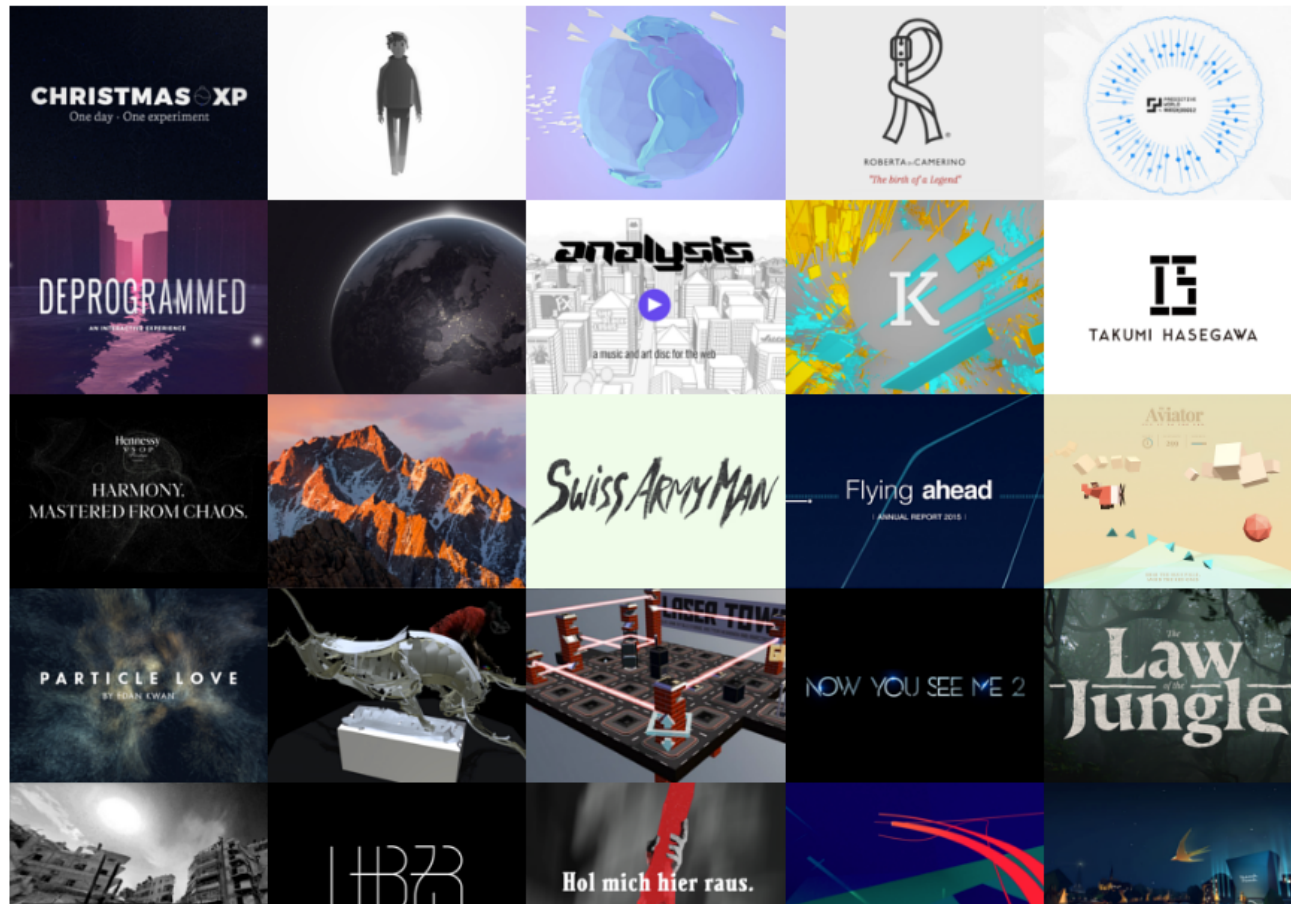
[documentation](#)
[examples](#)

[download](#)

[source code](#)
[questions](#)
[forum](#)
[chat](#)

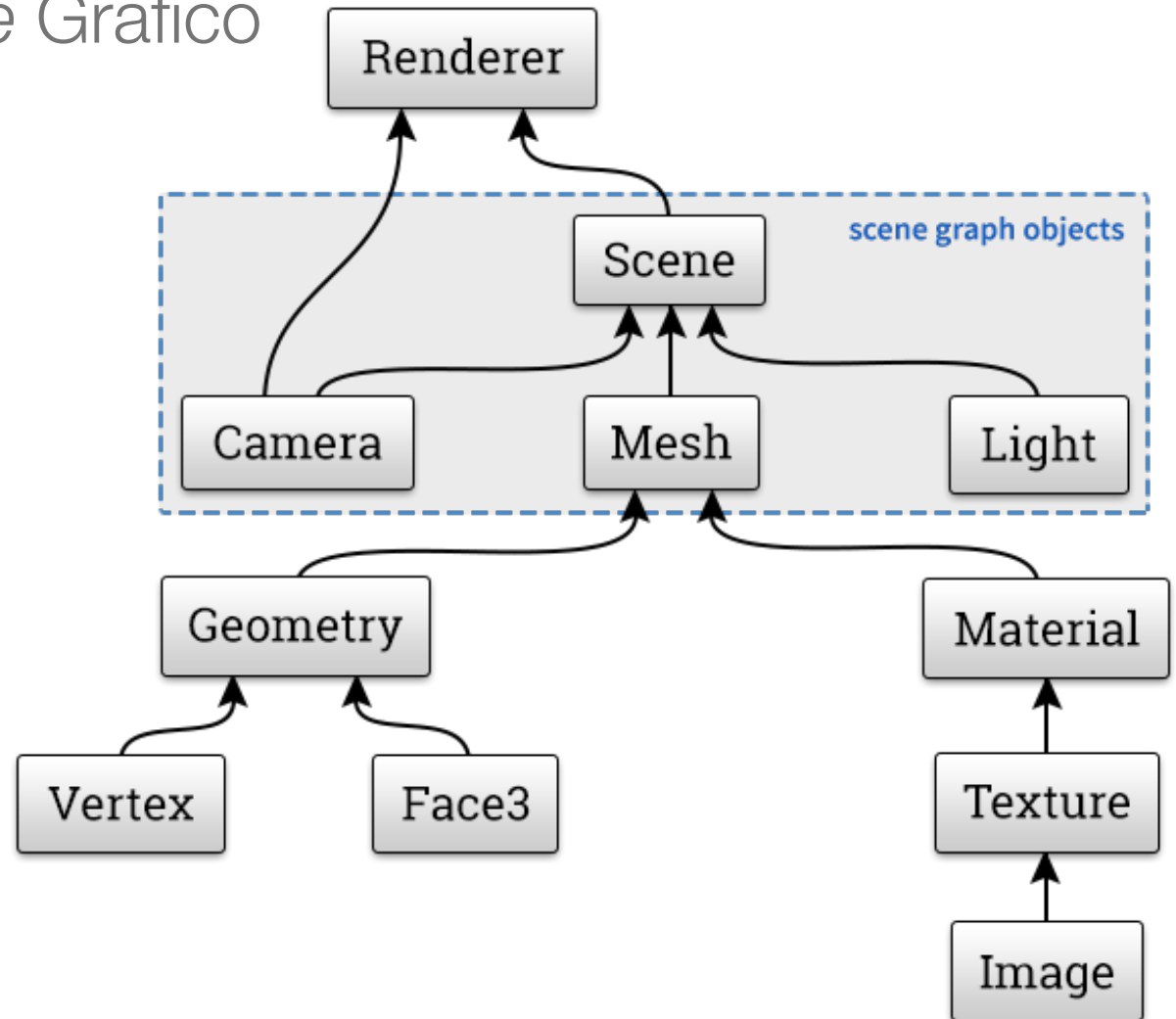
[editor](#)

Intro to WebGL
with Three.js



Three.js

- Prove abstrações para os principais processos vinculados ao *Pipeline* Gráfico
 - Grafo de Cena



Three.js

- Exemplo minimo (HTML)

```
<html>
  <head>
    <title>MATA65 - Computacao Grafica</title>
    <meta http-equiv="content-type" content="text/html; charset=UTF-8">

    <script type="text/javascript" src="../../libs/three.js"></script>
    <script type="text/javascript" src="6-SimpleThree.js"></script>
  </head>

  <body onload="init();" >
    <h1>Laboratorio 0 - Exemplo 6</h1><br />
    <p>Hello World em Three.js</p>
    <br/>
    <div id="WebGL-output" ></div>
    <br/>
  </body>
</html>
```

Three.js

- Exemplo minimo (JavaScript)

```
// Hello World em Three.js
function init() {
    var scene          = new THREE.Scene();
    var renderer = new THREE.WebGLRenderer();
    var camera         = new THREE.Camera();

    renderer.setClearColor(new THREE.Color(0.0, 0.0, 0.0));
    renderer.setSize(window.innerWidth*0.9,window.innerHeight*0.9);

    document.getElementById("WebGL-output").appendChild(renderer.domElement);

    renderer.render(scene, camera);
};
```

Three.js

- Desenhando uma geometria 2D simples (JavaScript)

```
// Desenhando uma Geometria 2D simples em Three.js
function init() {
    var scene = new THREE.Scene();
    var renderer = new THREE.WebGLRenderer();
    renderer.setClearColor(new THREE.Color(0.0, 0.0, 0.0));
    renderer.setSize(window.innerWidth*0.7, window.innerHeight*0.7);
    var camera = new THREE.OrthographicCamera( -1.0, 1.0, 1.0, -1.0, -1.0, 1.0 );
    scene.add( camera );
    var geometry = new THREE.Geometry();
    geometry.vertices.push(    new THREE.Vector3( -1.0, 0.5, 0 ),
                              new THREE.Vector3( -0.5, -0.5, 0 ),
                              new THREE.Vector3( 0, 0.5, 0 ),
                              new THREE.Vector3( 0.5, -0.5, 0 ),
                              new THREE.Vector3( 1.0, 0.5, 0 )
                              );

    var line = new THREE.Line(geometry);
    scene.add( line );
    document.getElementById("WebGL-output").appendChild(renderer.domElement);
    renderer.clear();
    renderer.render(scene, camera);
};
```

Three.js

- Desenhando uma geometria 2D simples com Material (JavaScript)

```
// Desenhando uma Geometria 2D simples com Material em Three.js
```

```
function init() {  
(...)  
  // Global Axis  
  var globalAxis = new THREE.AxisHelper( 1.0 );  
  scene.add( globalAxis );  
  var geometry = new THREE.Geometry();  
  geometry.vertices.push(    new THREE.Vector3( -1.0, 0.5, 0 ),  
                           new THREE.Vector3( -0.5, -0.5, 0 ),  
                           new THREE.Vector3( 0, 0.5, 0 ),  
                           new THREE.Vector3( 0.5, -0.5, 0 ),  
                           new THREE.Vector3( 1.0, 0.5, 0 )  
                           );  
  var line = new THREE.Line(geometry);  
  scene.add( line );  
}
```

Three.js

- Desenhando uma geometria 2D simples com Material (JavaScript) (cont.)

```
var geometry2 = new THREE.Geometry();
geometry2.vertices.push(new THREE.Vector3( -1.0, 0.75, 0 ),
                        new THREE.Vector3( -0.5, -0.25, 0 ),
                        new THREE.Vector3( 0, 0.75, 0 ),
                        new THREE.Vector3( 0.5, -0.25, 0 ),
                        new THREE.Vector3( 1.0, 0.75, 0 )
                        );

geometry2.colors.push( new THREE.Color(1.0, 1.0, 1.0),
                      new THREE.Color(1.0, 1.0, 0.0),
                      new THREE.Color(1.0, 0.0, 0.0),
                      new THREE.Color(0.0, 1.0, 0.0),
                      new THREE.Color(0.0, 1.0, 1.0)
                      );

var material = new THREE.LineBasicMaterial({color:0xff0000});
var line2 = new THREE.Line(geometry2, material);
scene.add( line2 );
```

Three.js

- Desenhando uma geometria 2D simples com Material (JavaScript) (cont.)

```
var Tmaterial = new THREE.LineBasicMaterial( {  
    linewidth: 2.0,  
    color: 0xffffffff,  
    vertexColors: THREE.VertexColors } );  
  
var line3 = new THREE.Line(geometry2, Tmaterial);  
line3.translateY(-0.5);  
  
scene.add( line3 );  
(...)  
};
```

Atividade 1

Atividade 1

- A atividade é individual e deve ser submetida como um arquivo compactado (somente formatos tgz, rar ou zip) contendo um diretório com o nome do aluno.
 - **Arquivos fora do padrão serão penalizados em 1,0 ponto.**
 - **O diretório compactado deve conter TODOS OS ARQUIVOS necessários para a correção/execução da sua atividade. Caso isso não ocorra a atividade NÃO SERÁ CORRIGIDA.**
- **As submissões só serão aceitas via MOODLE.**
- A cada dia de atraso será descontado 1,0 ponto da nota final.

A Seguir...

LAB 2: Objetos Geométricos