WebGL
Up and Running

Tony Parisi

http://www.tonyparisi.com/

Get the Code

git clone https://github.com/tparisi/WebGLBook/

svn co svn://iscene.com/svn/webglbook

http://iscene.com/WebGLBook/webglbook.zip

About the Example Content

- Models purchased from Turbosquid http://www.turbosquid.com
  - Do NOT redistribute
  - Before using in your own applications, join the service, download and/or purchase, and convert. Read TOS.
- Models from Blendswap http://www.blendswap.com/
  - CC licensed
  - Before using in your own applications, join the service, download yourself and covert. Read TOS.
- Images
  - Variously attributed; if you don't see attribution, please email me

About Me

- Serial entrepreneur
- Founder, stealth game startup
- Consulting architect and CTO
- Web3D co-creator, VRML and X3D
- Author, WebGL Up and Running (O'Reilly 2012)

Contact Information
tparisi@gmail.com
Skype: auradeluxe
http://twitter.com/auradeluxe
http://www.tonyparisi.com/

Get the book!
eBook/Early Release: May 22, 2012
Print version: July 2012
http://shop.oreilly.com/product/0636920024729.do
Discount Code: WEBGLPRE

Agenda

1. An Introduction to WebGL

The New 3D API Standard

- OpenGL ES™ in a browser
- JavaScript API bindings
- Supported in ***nearly*** all modern browsers
- Supported on many devices
- Shipped since early 2011

…and it's Awesome.

WebGL - A Technical Definition

WebGL is a royalty-free, cross-platform API that brings OpenGL ES 2.0 to the web as a 3D drawing context within HTML, exposed as low-level Document Object Model interfaces. It uses the OpenGL shading language, GLSL ES, and can be cleanly combined with other web content that is layered on top or underneath the 3D content. It is ideally suited for dynamic 3D web applications in the JavaScript programming language, and will be fully integrated in leading web browsers.

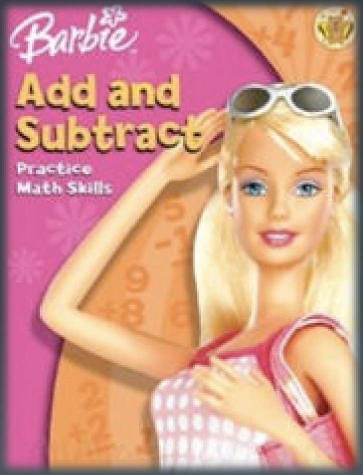
http://www.khronos.org/webgl/

Deconstructing WebGL

- WebGL is an API
  - No file format, no DOM
  - Uses new kind of Canvas element drawing context
- WebGL is based on OpenGL ES 2.0
  - It's what's in your phone
- WebGL combines with other web content
  - Integrates seamlessly with other stuff on the page
- WebGL is built for dynamic web applications
  - Runtime is the web browser, resources are URLs
- WebGL is cross-platform
- WebGL is royalty-free

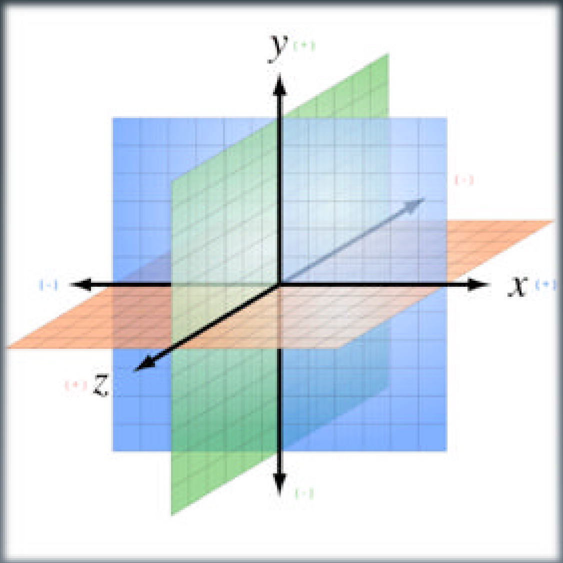
3D Graphics in Four Pages: A Primer

“Math is hard… I like shopping!”
-- Barbie



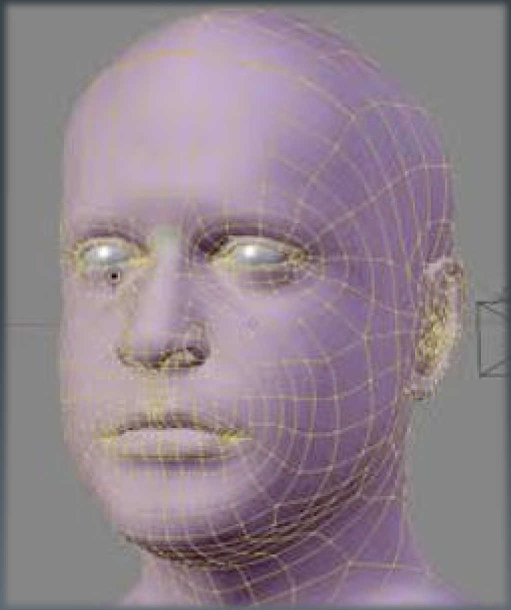
3D Coordinate Systems

- Similar to (2D) Canvas coordinate system
- Adds z coordinate (not same as CSS z-ordering)



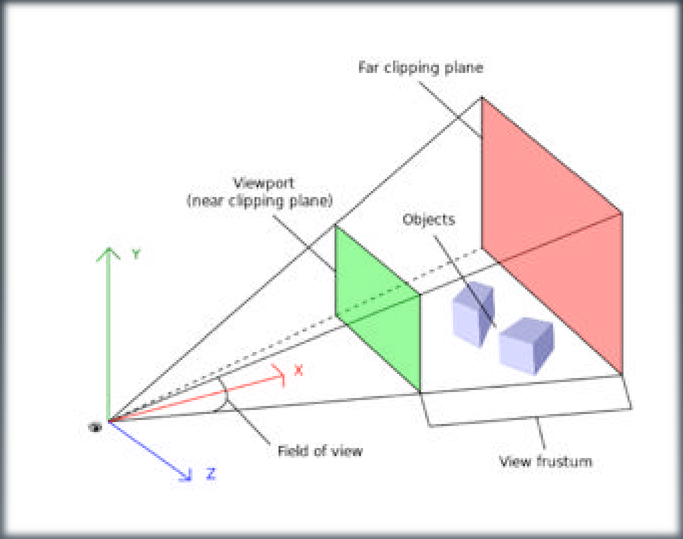
Representing Visuals

- Meshes - 3D shapes composed of polygons (three or more vertices); the most common type of rendered 3D object
- Materials - define surface properties (color, shininess, transparency, shading)
- Texture Maps - bitmaps applied to an object's surface
- Lights - illuminate scenes and shade objects



Transforms, Matrices, Cameras and Viewports

- Before drawing, objects can be *transformed* (moved) in x, y, z
- Transforms *compose* – are inherited from ancestors
- Transforms are implemented using *matrices*
- 3D information is *projected* onto a 2D *viewport*
- The *camera* defines projection
- Projections also use matrices



Shaders

- Programs written in a high-level C-like language (GLSL ES)
- Originally developed to implement highly realistic effects
- Required for WebGL development

The WebGL API

The Anatomy of a WebGL Application

- Create a Canvas element
- Obtain a drawing context
- Initialize the viewport
- Create one or more buffers
- Create one or more matrices
- Create one or more shaders
- Initialize the shaders
- Draw one or more primitives

Code
Chapter 1/Example 1-1.html

The Bottom Line

- A lot of work – very low-level API
- Amazing power - you can do almost anything down to the hardware
- Frameworks, toolkits, libraries will be a big help

2. Your First WebGL Program

Three.js – A JavaScript 3D Engine

- Represents WebGL at a high level using standard 3D graphics concepts
- Feature Rich – math, graphics, animation, interaction, file loaders
- Object-oriented
- Renders to 3D WebGL or 2D standard Canvas
- Fast, robust, well-maintained
- Chock full of examples

- Not a game engine
- Not an application framework

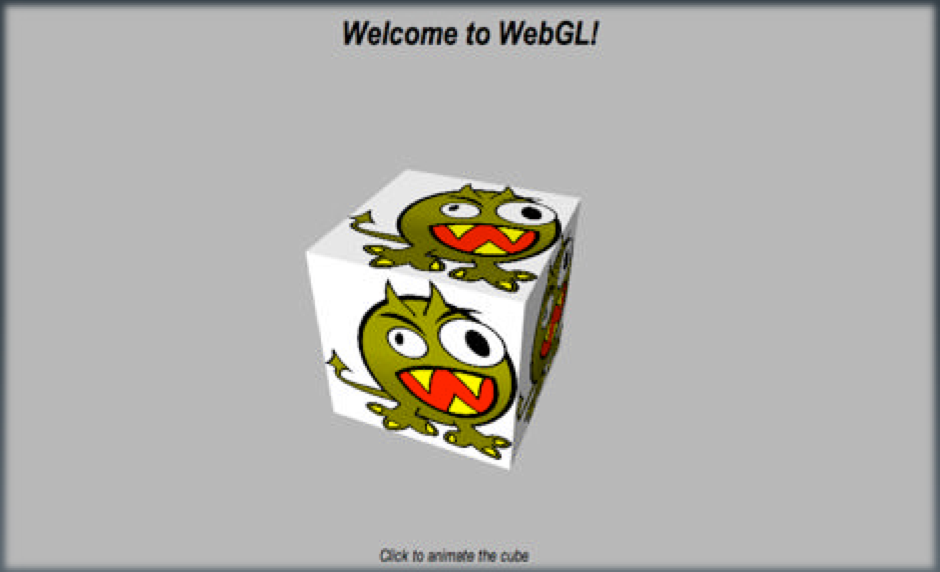
https://github.com/mrdoob/three.js/

The Square, Revisited

Here’s That Square Again.

Code
Chapter 2/Example 2-1.html

A Real Example



Code
Chapter 2/Example 2-2.html

Using a Light to Shade the Scene

```
// Create a directional light to show off the object
      var light = new
        THREE.DirectionalLight( 0xffffff, 1.5);
        light.position.set(0, 0, 1);
        scene.add( light );
```

Code
Chapter 2/Example 2-2.html

Creating a Shaded, Texture-Mapped Cube

```
// Create a shaded, texture-mapped cube and add it to the scene
// First, create the texture map
var mapUrl = "../images/molumen_small_funny_angry_monster.jpg";
var map = THREE.ImageUtils.loadTexture(mapUrl);

// Now, create a Phong material to show shading; pass in the map
var material = new THREE.MeshPhongMaterial({ map: map });

// Create the cube geometry
var geometry = new THREE.CubeGeometry(1, 1, 1);

// And put the geometry and material together into a mesh
cube = new THREE.Mesh(geometry, material);
```

Code

Chapter 2/Example 2-2.html

Rotating the Object

```
// Turn it toward the scene, or we won't see the cube shape!
cube.rotation.x = Math.PI / 5;
cube.rotation.y = Math.PI / 5;
```

Code

Chapter 2/Example 2-2.html

The Run Loop and `requestAnimationFrame()`

```
function run()
{
        // Render the scene
        renderer.render( scene, camera );

        // Spin the cube for next frame
        if (animating)
        {
                cube.rotation.y -= 0.01;
        }

        // Ask for another frame
        requestAnimationFrame(run);
}
```

Code
Chapter 2/Example 2-2.html

Handling the Mouse

```
function addMouseHandler()
{
        var dom = renderer.domElement;

        dom.addEventListener( 'mouseup', onMouseUp,
false);
}

function onMouseUp    (event)
{
        event.preventDefault();

        animating = !animating;
}
```

Code

Chapter 2/Example 2-2.html

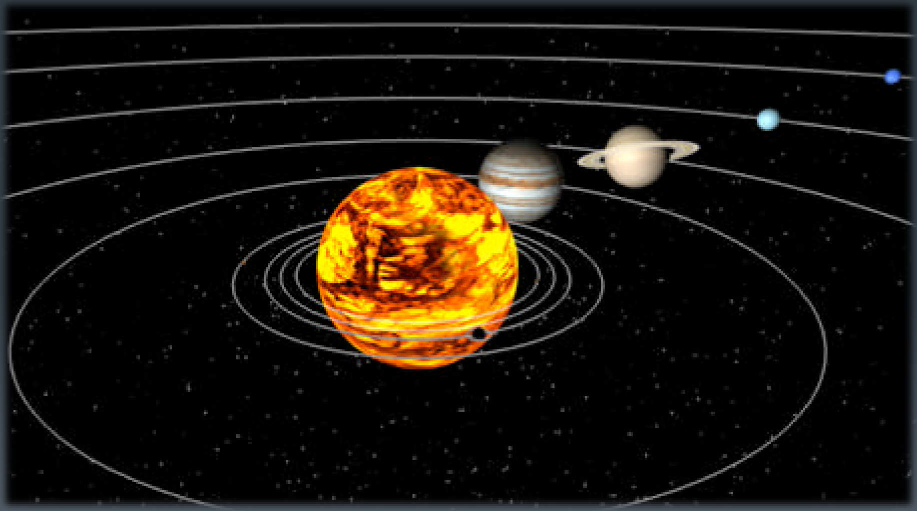
Review

- Three.js gets us going in 40 lines instead of 200
- Create an HTML page, easily add 3D content
- Pretty it up with textures and shading
- Bring it to life with animation
- Make it interactive with DOM mouse handler

We're Up and Running!

3. Graphics

Let’s Build a Solar System



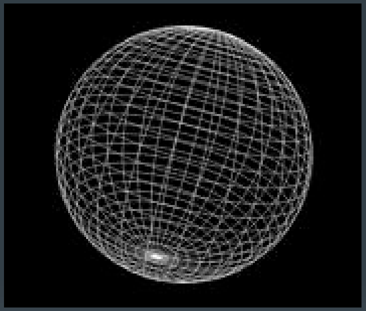
Code
Chapter 3/graphics-solar-system.html

Sim.js – A Simple Simulation Framework for WebGL/Three.js

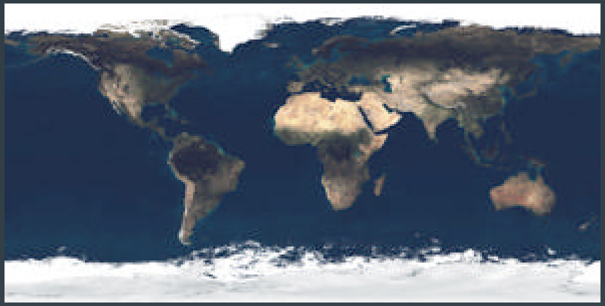
- Wraps common Three.js setup and teardown
- Implements the run loop
- Sets up handlers for mouse and keyboard DOM Events
- Provides foundation objects (Application, Base Object and PubSub)
- Handles browser detection and context lost events
- Coming soon to Github

  https://github.com/tparisi/Sim.js

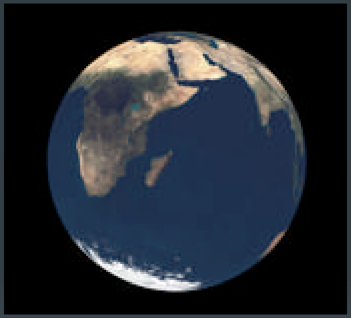
Creating Meshes



+



=



Code

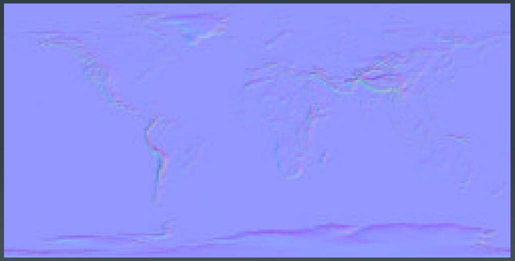
Chapter 3/graphics-earth-basic.html

Using Materials, Textures, Lights and Shaders

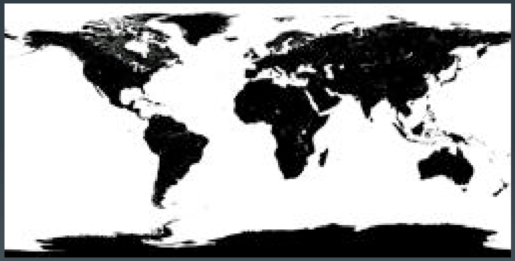
Code
Chapter 3/graphics-earth-shader.html

The Texture Maps

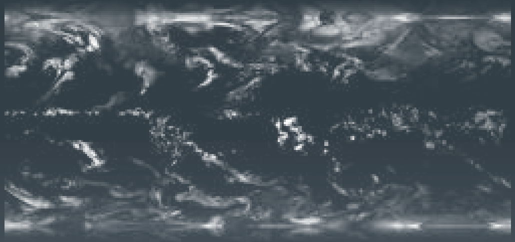
Normal Map (Elevation)



Specular Map (Highlights)

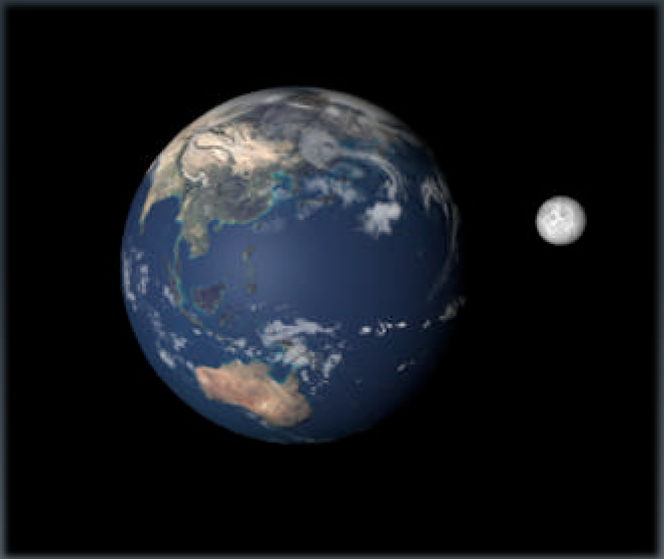


Cloud Layer (Transparency)



Code
Chapter 3/graphics-earth-shader.html

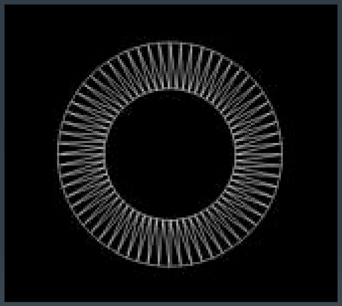
Building a Transform Hierarchy



Code

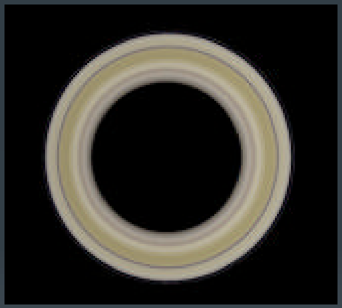
Chapter 3/graphics-earth-moon.html

Creating Custom Geometry



+

=



Code

Chapter 3/saturn.js

Rendering Points and Lines

- Points and lines are API drawing primitives in WebGL
- Three.js has built-In point and line support
  - `THREE.ParticleSystem, THREE.ParticleBasicMaterial`
  - `THREE.Line, THREE.LineBasicMaterial`

Code

Chapter 3/stars.js, Chapter 3/orbit.js

Writing a Shader

- Put GLSL ES source in a script element, JavaScript string variable or external text file
- Use Three.js `ShaderMaterial` object for your material
- Update `uniform` parameter values during run loop

Code
Chapter 3/graphics-solar-system.html

The Vertex Shader

```
<script id="vertexShader" type="x-shader/x-vertex">

        varying vec2 texCoord;

        void main()
        {
                texCoord = uv;
                vec4 mvPosition = modelViewMatrix * vec4( position, 1.0 );
                gl_Position = projectionMatrix * mvPosition;
        }

</script>
```

Code

Chapter 3/graphics-solar-system.html

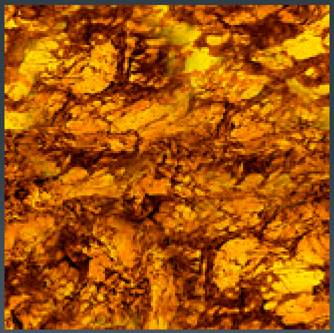
The Fragment Shader

```
<script id="fragmentShader" type="x-shader/x-fragment">

          uniform float time;

          uniform sampler2D texture1;
          uniform sampler2D texture2;

          varying vec2 texCoord;

          void main( void ) {
                     vec4 noise = texture2D( texture1, texCoord );

                     vec2 T1 = texCoord + vec2( 1.5, -1.5 ) * time  * 0.01;
                     vec2 T2 = texCoord + vec2( -0.5, 2.0 ) * time *  0.01;

                     T1.x -= noise.r * 2.0;
                     T1.y += noise.g * 4.0;
                     T2.x += noise.g * 0.2;
                     T2.y += noise.b * 0.2;

                     float p = texture2D( texture1, T1 * 2.0 ).a + 0.25;

                     vec4 color = texture2D( texture2, T2 );
                     vec4 temp = color * 2.0 * ( vec4( p, p, p, p ) ) + ( color * color );
                     gl_FragColor = temp;
          }

</script>
```

Code
Chapter 3/graphics-solar-system.html

Texture Maps for the Shader

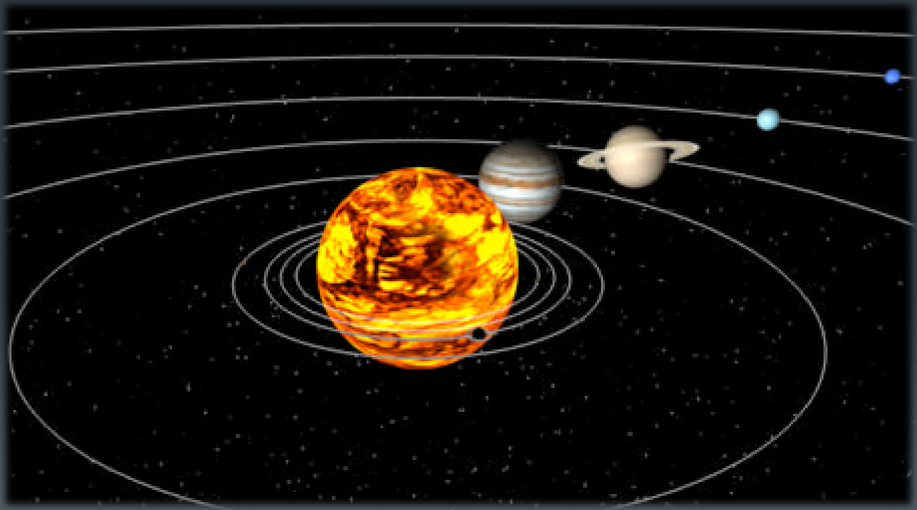
Color Map



Noise/Alpha Map

Code
Chapter 3/graphics-solar-system.html

Another Look



Code
Chapter 3/graphics-solar-system.html

Review

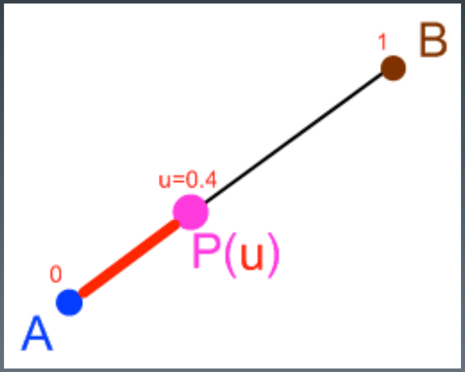
- Creating meshes
- Shading, materials, textures and lights
- Building a transform hierarchy
- Creating custom geometry
- Rendering points and lines
- Writing a shader

4. Animation

Animation Basics

- Frame-based vs. Time-based
- Interpolation and Tweening
- Key Frames
- Articulated Animation

Interpolation



Sample Key Frame Data

```
var keys = [0, 0.25, 1];
var values = [new THREE.Vector3(0, 0, 0),
        new THREE.Vector3(0, 1, 0),
        new THREE.Vector3(0, 2, 5)];
```

Creating a Tween

```
MovingBall.prototype.animate = function()
{
    var newpos;
    if (this.object3D.position.x > 0)
    {
        newpos = this.object3D.position.x - 6.667;
    }
    else
    {
        newpos = this.object3D.position.x + 6.667;
    }

    new TWEEN.Tween(this.object3D.position)
    .to( {
        x: newpos
    }, 2000).start();
}
```

Code

Chapter 4/tween-basic.html

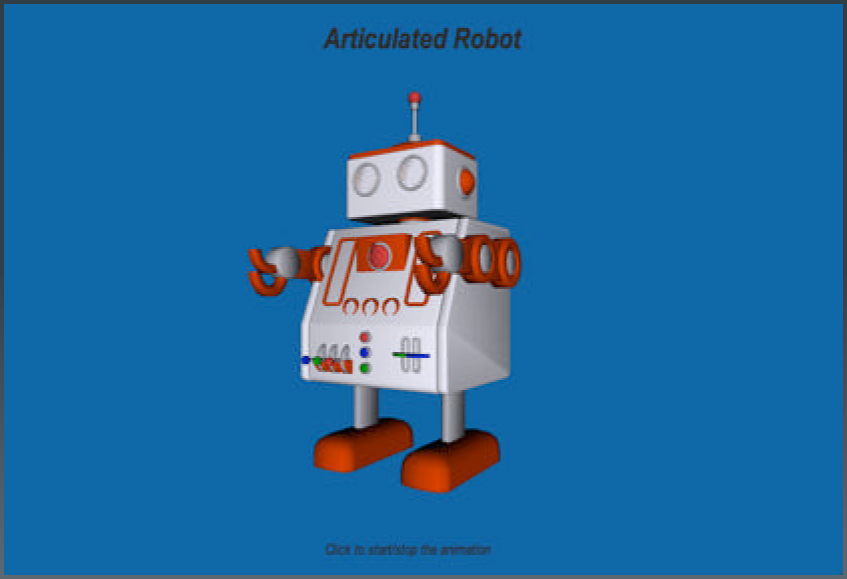
Tweens with Easing

```
MovingBall.prototype.animate = function()
{
    var newpos, easefn;
    if (this.object3D.position.y > 0)
    {
        newpos = this.object3D.position.y - 6.667;
        easefn = MovingBall.useBounceFunction ?
                TWEEN.Easing.Bounce.EaseOut :
                TWEEN.Easing.Quadratic.EaseOut;
    }
    else
    {
        newpos = this.object3D.position.y + 6.667;
        easefn = MovingBall.useBounceFunction ?
                TWEEN.Easing.Bounce.EaseIn :
                TWEEN.Easing.Quadratic.EaseIn;
    }

    new TWEEN.Tween(this.object3D.position)
    .to( {
        y: newpos
    }, 2000)
    .easing(easefn).start();

}
```

Code

Chapter 4/tween-easing.html

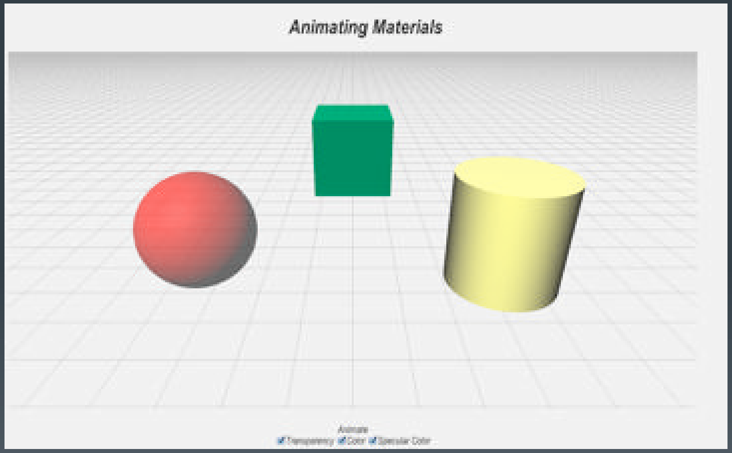
Animating an Articulated Model with Key Frames

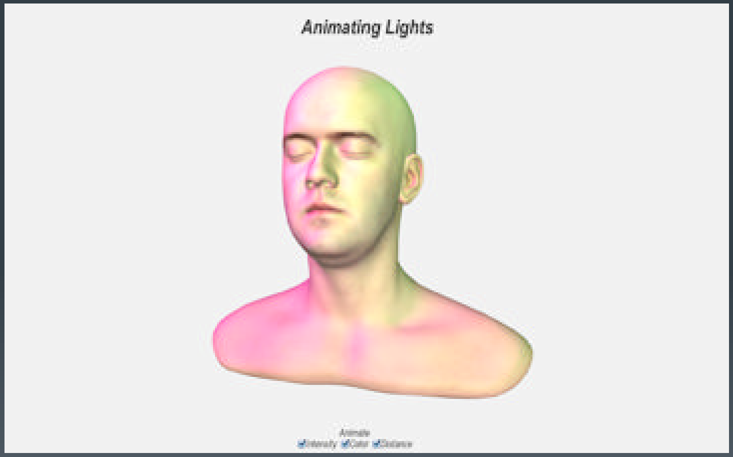


Code

Chapter 4/keyframe-robot.html

Animating Materials and Lights

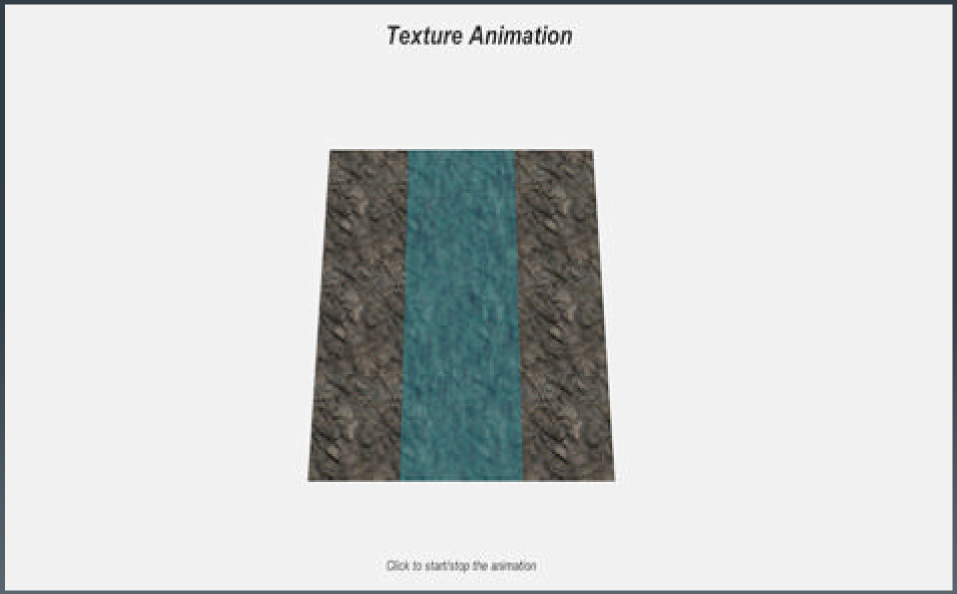




Code

Chapter 4/keyframe-material.html, Chapter 4/keyframe-lights.html

Animating Textures



Code
Chapter 4/keyframe-texture.html

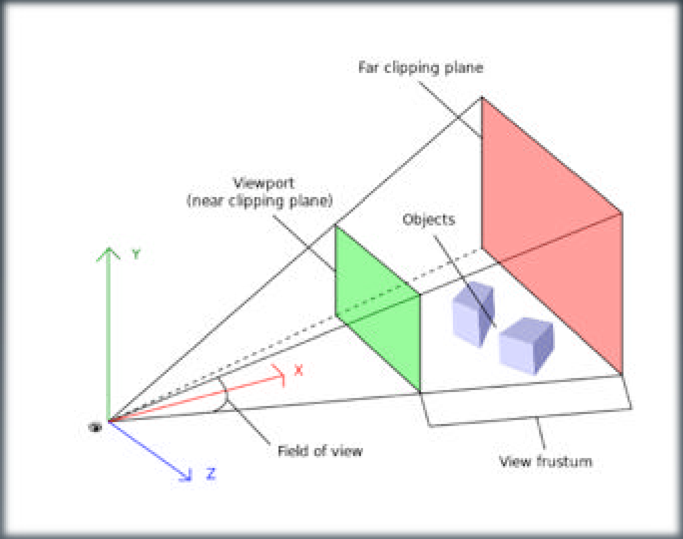
Review

- Basic animation concepts
- Creating tweens
- Animating an articulated model with key frames
- Animating materials and lights
- Animating textures

5. Interaction

Hit Detection, Picking and Projection

- WebGL has no built-in hit detection
- Three.js has hit detection helper – it does the inverse of 3D->2D projection to find object under the mouse



Implementing Rollovers

```
Control.prototype.handleMouseOver = function(x, y)
{
    this.mesh.scale.set(1.05, 1.05, 1.05);
    this.mesh.material.ambient.setRGB(.777,.777,.777);
}

Control.prototype.handleMouseOut = function(x, y)
{
    this.mesh.scale.set(1, 1, 1);
    this.mesh.material.ambient.setRGB(.667, .667, .
667);
}
```

Code
Chapter 5/interaction-simple.html

Implementing Clicks

```
Control.prototype.handleMouseDown = function(x, y, hitPoint)
{
    if (!this.selected)
    {
        this.select();
    }
    else
    {
        this.deselect();
    }
}

Control.prototype.select = function()
{
    if (!this.savedposition)
    {
        this.savedposition = this.mesh.position.clone();
    }

    new TWEEN.Tween(this.mesh.position)
    .to({
        x : 0,
        y : 0,
        z: 2
        }, 500).start();

    this.selected = true;
    this.publish("selected", this, true);
}
```

Code

Chapter 5/interaction-simple.html

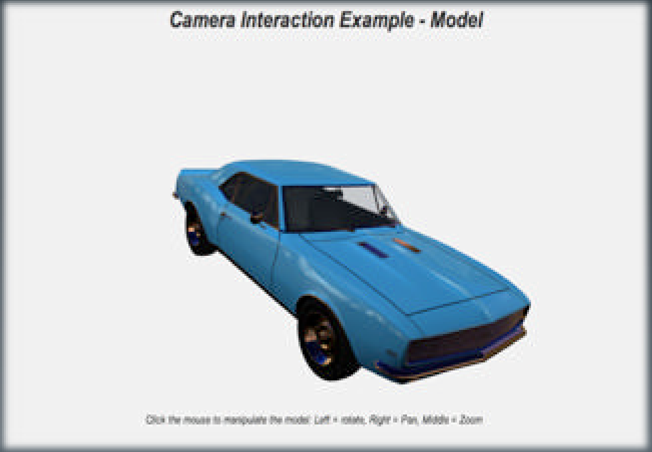
Implementing Dragging

- PlaneDragger class
  - Uses `THREE.Projector` to find hit point in scene
  - Pick against invisible plane geometry
- Use Tween with drag to make UI feel more natural and intuitive

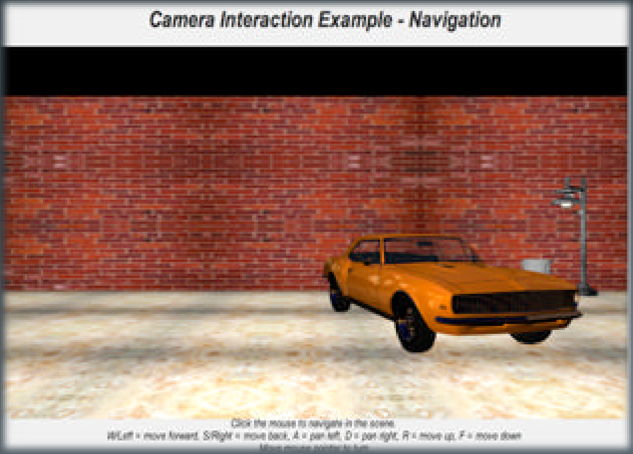
Code
Chapter 5/interaction-drag.html

Camera-Based Interaction

Trackball Camera for Model Viewing


Camera Interaction Example - Model
Click the mouse to manipulate the model: Left = rotate, Right = Pan, Middle = Zoom

First-Person Camera for Scene Navigation


Camera Interaction Example - Navigation
Click the mouse to navigate in the scene.
W/Left = move forward, S/Right = move back, A = pan left, D = pan right, R = move up, F = move down
Move mouse pointer to turn

Code

Chapter 5/interaction-camera-model.html, Chapter 5/interaction-camera-navigation.html

Review

- Hit detection, picking and projection
- Implementing rollovers and clicks
- Implementing dragging
- Camera-based interaction

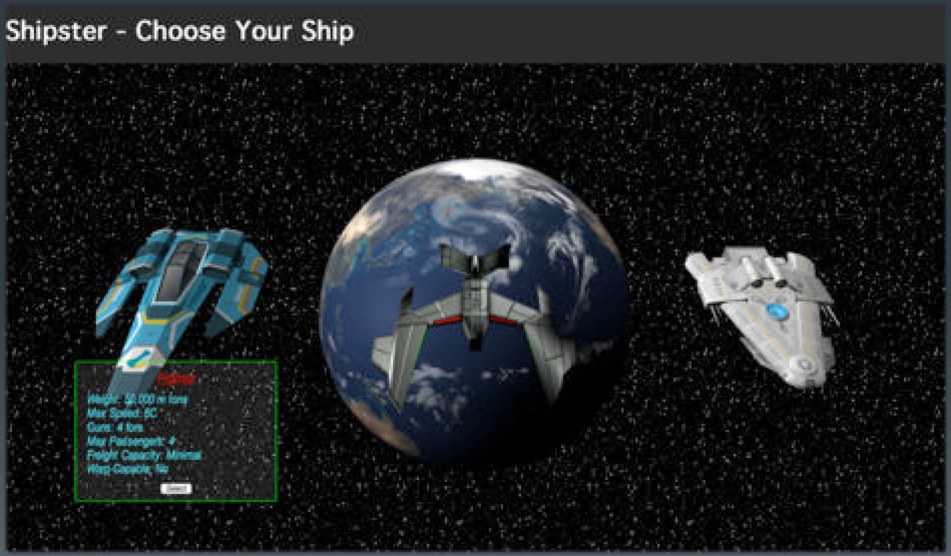
6. Integrating 2D and 3D

WebGL’s Secret Weapon

WebGL is a royalty-free, cross-platform API that brings OpenGL ES 2.0 to the web as a 3D drawing context within HTML, exposed as low-level Document Object Model interfaces. It uses the OpenGL shading language, GLSL ES, ***and can be cleanly combined with other web content that is layered on top or underneath the 3D content****. It is ideally suited for dynamic 3D web applications in the JavaScript programming language, and will be fully integrated in leading web browsers.*

- ✓Breaks down window boundaries
- ✓2D overlaid on 3D
- ✓3D overlaid on 2D
- ✓Canvas2D as a texture
- ✓Video as a texture
- ✓The ultimate mashup technology

Combining Dynamic HTML and WebGL


Shipster - Choose Your Ship

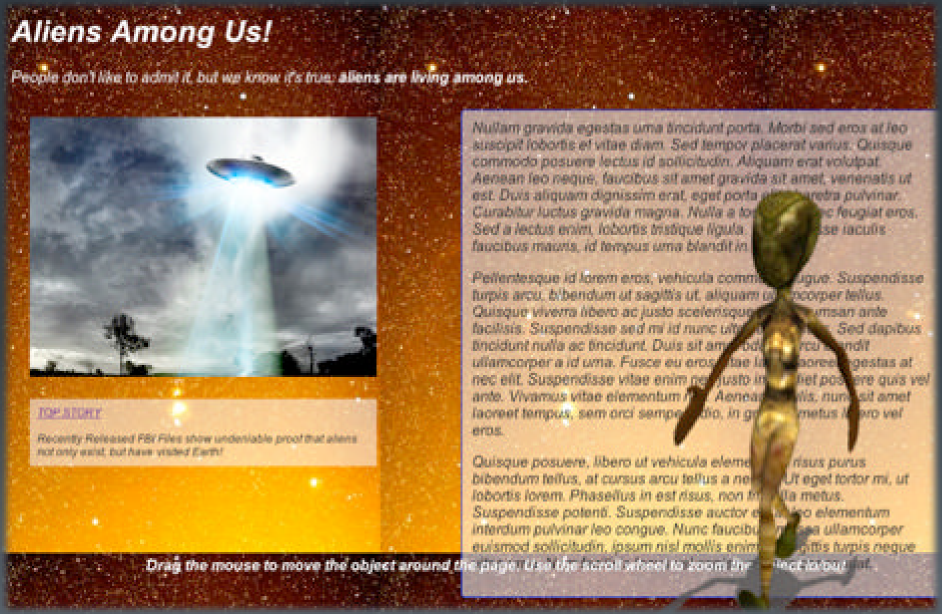
Fighter

Weight: 50,000 m tons
Max Speed: 8C
Guns: 4 fore
Max Passengers: 4
Freight Capacity: Minimal
Warp Capable: No

Code

Chapter 6/integration-div.html

Overlaying 3D Visuals on 2D Pages



Aliens Among Us!

*People don't like to admit it, but we know it's true: **aliens are living among us.***

TOP STORY

Recently Released FBI Files show undeniable proof that aliens not only exist, but have visited Earth!

Nullam gravida egestas urna tincidunt porta. Morbi sed eros at leo suscipit lobortis et vitae diam. Sed tempor placerat varius. Quisque commodo posuere lectus id sollicitudin. Aliquam erat volutpat. Aenean leo neque, faucibus sit amet gravida sit amet, venenatis ut est. Duis aliquam dignissim erat, eget porta [illegible]retra pulvinar. Curabitur luctus gravida magna. Nulla a to[illegible]c feugiat eros. Sed a lectus enim, lobortis tristique ligula [illegible]se iaculis faucibus mauris, id tempus urna blandit in.

Pellentesque id lorem eros, vehicula comm[illegible]gue. Suspendisse turpis arcu, bibendum ut sagittis ut, aliquam u[illegible]corper tellus. Quisque viverra libero ac justo scelerisque[illegible]umsan ante facilisis. Suspendisse sed mi id nunc ult[illegible] Sed dapibus tincidunt nulla ac tincidunt. Duis sit am[illegible]ndit ullamcorper a id urna. Fusce eu eros[illegible]gestas at nec elit. Suspendisse vitae enim [illegible]justo i[illegible]bet posuere quis vel ante. Vivamus vitae elementum [illegible] Aenea[illegible]llis, nun[illegible]sit amet laoreet tempus, sem orci semper [illegible]dio, in g[illegible]metus [illegible]ro vel eros.

Quisque posuere, libero ut vehicula eleme[illegible]risus purus bibendum tellus, at cursus arcu tellus a ne[illegible] Ut eget tortor mi, ut lobortis lorem. Phasellus in est risus, non [illegible]lla metus. Suspendisse potenti. Suspendisse auctor e[illegible]o elementum interdum pulvinar leo congue. Nunc faucibu[illegible]a ullamcorper euismod sollicitudin, ipsum nisl mollis enim[illegible]ttis turpis neque

Drag the mouse to move the object around the page. Use the scroll wheel to zoom the [illegible]ect in/out.

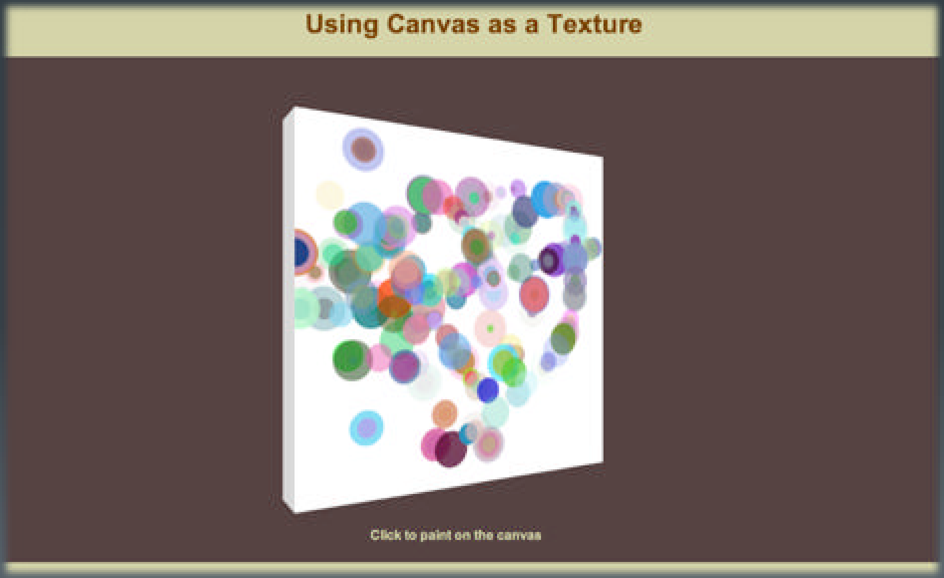
http://www.tonyparisi.com/

Code

Chapter 6/integration-object.html

May 23, 2012
WebGL Up and Running

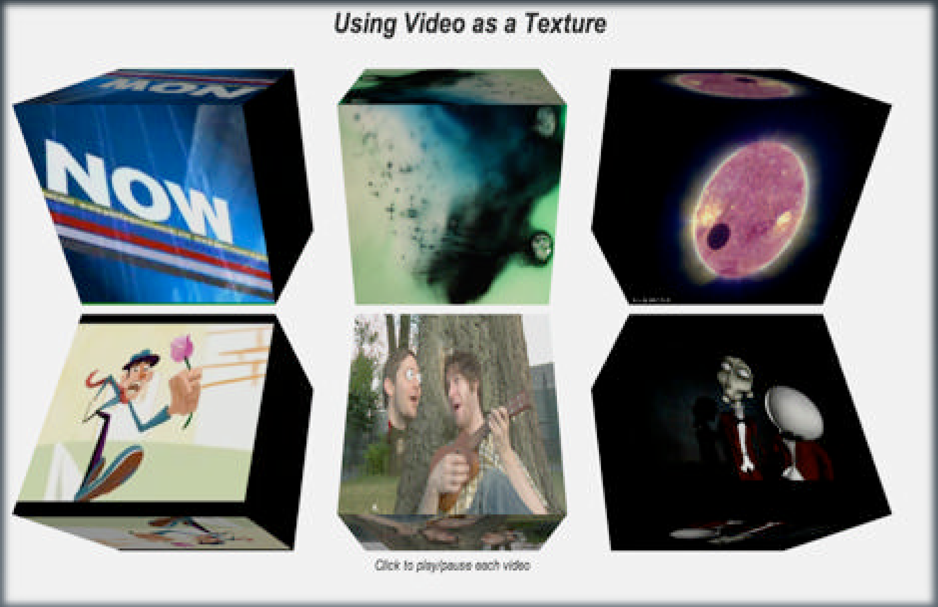
Creating Dynamic Textures with Canvas 2D



Code

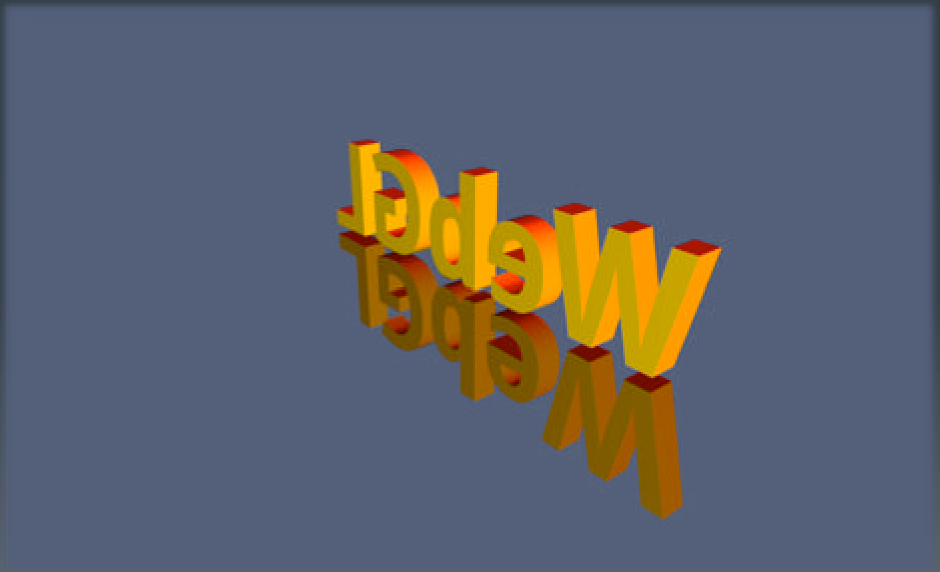
Chapter 6/integration-canvas.html

Using Video as a Texture



Code
Chapter 6/integration-video.html

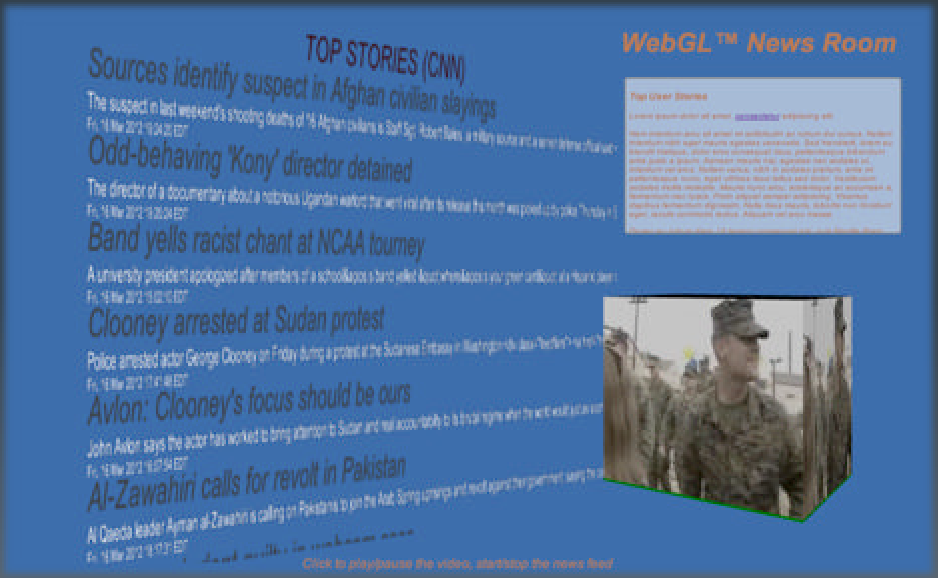
Rendering Dynamically Generated 3D Text



Code

Chapter 6/integration-text.html

WebGL for Ultimate Mashups



TOP STORIES (CNN)

Sources identify suspect in Afghan civilian slayings

The suspect in last weekend's shooting deaths of 16 Afghan civilians is Staff Sgt. Robert Bales, a military source and a senior defense official said

Fri, 16 Mar 2012 19:24:20 EDT

Odd-behaving 'Kony' director detained

The director of a documentary about a notorious Ugandan warlord that went viral after its release this month was posed to by police Thursday in

Fri, 16 Mar 2012 19:20:24 EDT

Band yells racist chant at NCAA tourney

A university president apologized after members of a school&apos;s band yelled &quot;where&apos;s your green card&quot; at a Puerto Rican

Fri, 16 Mar 2012 15:02:10 EDT

Clooney arrested at Sudan protest

Police arrested actor George Clooney on Friday during a protest at the Sudanese Embassy in Washington

Fri, 16 Mar 2012 17:41:48 EDT

Avlon: Clooney's focus should be ours

John Avlon says the actor has worked to bring attention to Sudan and real accountability to its brutal regime

Fri, 16 Mar 2012 16:07:54 EDT

Al-Zawahiri calls for revolt in Pakistan

Al Qaeda leader Ayman al-Zawahiri is calling on Pakistanis to join the Arab Spring uprisings and revolt against their government

Fri, 16 Mar 2012 18:17:31 EDT

Click to play/pause the video, start/stop the news feed

Code

Chapter 6/integration-media.html

Review

- Combining Dynamic HTML and WebGL
- Overlaying 3D visuals on 2D Pages
- Creating dynamic textures with Canvas 2D
- Using video as a texture
- Rendering dynamically generated 3D text
- WebGL for ultimate mashups

The tyranny of the rectangle is officially over.
Multimedia developers of the world, unite!
You have nothing to lose… but window borders!

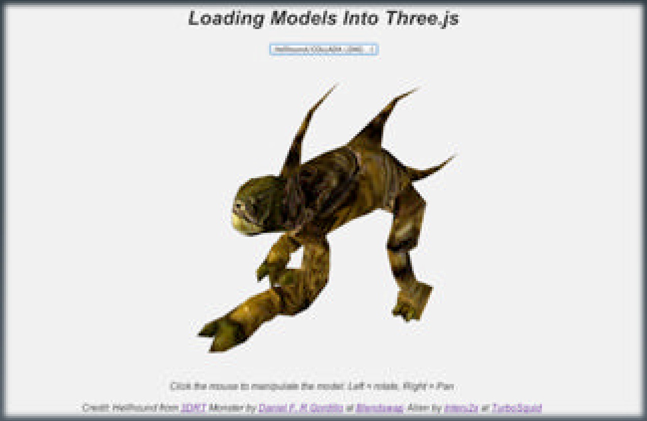
7. WebGL in Production

Choosing a Framework

- Open Source rendering engines/frameworks
  - Three.js
  - CubicVR
  - SceneGL
  - GLGE
- Emerging game engines
  - Gladius
  - KickJS
  - Skybox
- Roll your own?

Loading 3D Content

- WebGL has no built-in file format
- Three.js loads several types
  - COLLADA
  - Three.js JSON format
  - Three.js binary format
  - Three.js JSON scene format


Loading Models Into Three.js

Click the mouse to manipulate the model. Left = rotate, Right = Pan

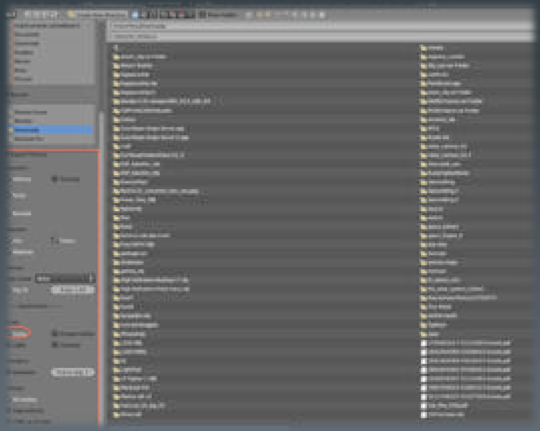
Credit: Hellhound from 3DRT Monster by Daniel F. R. Gordillo at Blendswap, Alien by [illegible] at TurboSquid

Code
Chapter 7/production-loader-model.html

Creating 3D Content

- Exporting art from blender
- Converting Wavefront OBJ to Three.js
- Other file format converters

```
python <path-to-three.js>/utils/
exporters/convert_obj_three.py -i
alien2.obj -o alien2.js
```



Loading a Three.js JSON Scene

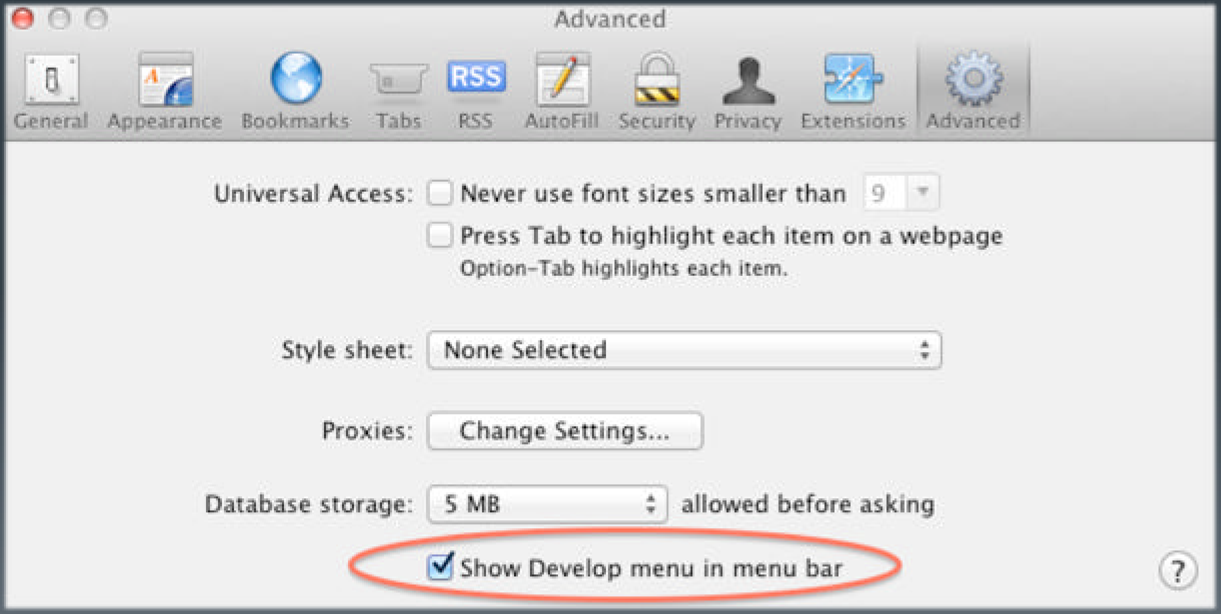
Browser Realities

- The elephant in the room: Not supported in IE
- Detecting WebGL support in your browser

```
var canvas = document.createElement("canvas");

var gl = null;
var msg = "Your browser does not support WebGL.";
try
{
  gl = canvas.getContext("experimental-webgl");
}
catch (e)
{
  msg = "Error creating WebGL Context!: " + e.toString();
}
if (!gl)
{
  throw new Error(msg);
}
```

Code
sim/webGLDetector.js

Turning WebGL on in Safari



Universal Access: ☐ Never use font sizes smaller than 9

☐ Press Tab to highlight each item on a webpage

Option-Tab highlights each item.

Style sheet: None Selected

Proxies: Change Settings...

Database storage: 5 MB allowed before asking

☑ Show Develop menu in menu bar

Handling Context Lost Events

```
ContextApp.prototype.handleContextLost = function(e)
{
          this.restart();
}

ContextApp.prototype.addContextListener = function()
{
          var that = this;

          this.renderer.domElement.addEventListener("webglcontextlost",
                              function(e) {
                                        that.handleContextLost(e);
                                        },
                              false);
}
```

Code

Chapter 7/production-context-lost.html

WebGL and Security

- Security holes in early implementations – now fixed
- Khronos, WebGL implementers and hardware manufacturers have done extensive security work
- Loading cross-origin resources
  - Use a server-side proxy
  - Use JSONP (not supported by all services)
  - Server that supports CORS headers (still new, support not widespread)

Code
Chapter 7/production-crossdomain-CORS.html

Review

- Choosing a framework
- Loading 3D content
- Creating 3D content
- Browser realities
- Handling context lost events
- WebGL and security

8. Your First WebGL Game

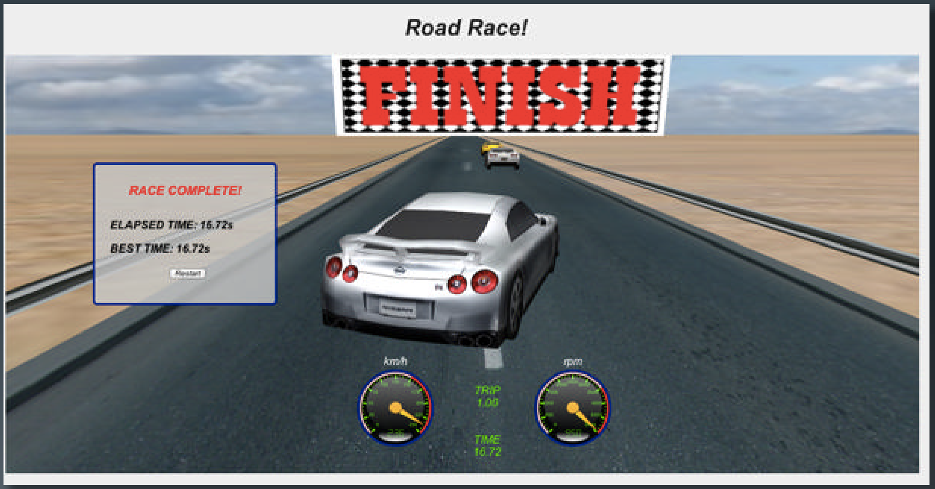
Let’s Build a Racing Game

Review Topics

- Graphics
- Models
- Animation
- 2D/3D integration
- Building a robust application

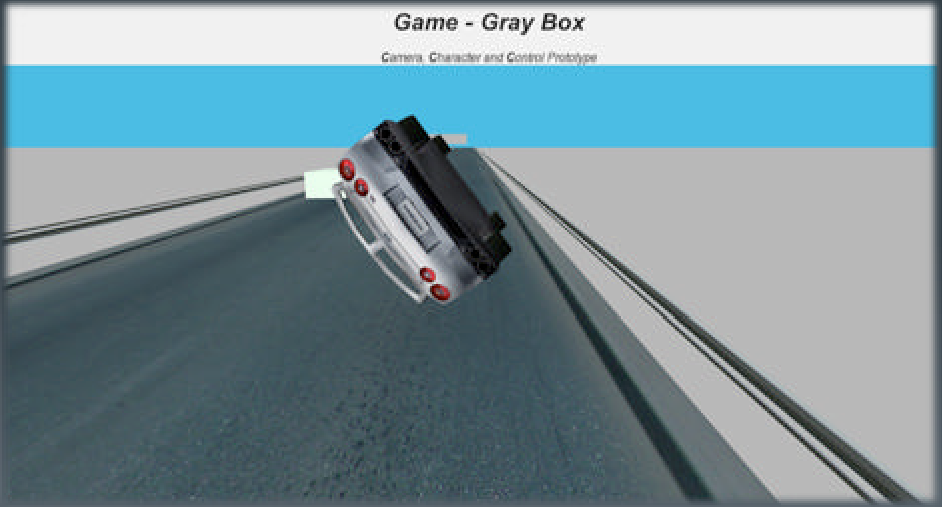
New Topics

- Creating a particle effect
- Keyboard input
- Sound



Camera, Character and Control

The "Gray Box" Prototype



Code

Chapter 8/game-graybox.html

Keyboard Controls

- Keyboard handling is DOM-based, like any Web app
- Sim.App defines key codes (see http://www.javascripter.net/faq/keycodes.htm)

```
/* key codes
37: left
38: up
39: right
40: down
*/
Sim.KeyCodes = {};
Sim.KeyCodes.KEY_LEFT  = 37;
Sim.KeyCodes.KEY_UP  = 38;
Sim.KeyCodes.KEY_RIGHT  = 39;
Sim.KeyCodes.KEY_DOWN  = 40;
```

- Game object defines `onKeyUp()`, `onKeyDown()`, passes it to `Player` object

Code
Chapter 8/game-graybox.html

Updating the Camera

```
Player.prototype.updateCamera = function()
{
		var camerapos = new THREE.Vector3(Player.CAMERA_OFFSET_X,
Player.CAMERA_OFFSET_Y, Player.CAMERA_OFFSET_Z);
		camerapos.addSelf(this.object3D.position);
		this.camera.position.copy(camerapos);
		this.camera.lookAt(this.object3D.position);

		// Rotate particle system to view-aligned to avoid nasty alpha sorting artifacts
		if (this.exhaust1)
		{
			this.exhaust1.object3D.rotation.x = this.camera.rotation.x;
		}

		if (this.exhaust2)
		{
			this.exhaust2.object3D.rotation.x = this.camera.rotation.x;
		}

}
```

Code
Chapter 8/game-graybox.html

Key Frame Animations

```
Car.crashPositionKeys = [0, .25, .75, 1];
Car.crashPositionValues = [ { x : -1, y: 0, z : 0},
                        { x: 0, y: 1, z: -1},
                        { x: 1, y: 0, z: -5},
                        { x : -1, y: 0, z : -2}
                        ];
Car.crashRotationKeys = [0, .25, .5, .75, 1];
Car.crashRotationValues = [ { z: 0, y: 0 },
                                { z: Math.PI, y: 0},
                                { z: Math.PI * 2, y: 0},
                                { z: Math.PI * 2, y: Math.PI},
                                { z: Math.PI * 2, y: Math.PI * 2},
                                ];

Car.crash_animation_time = 2000;
```

Code
Chapter 8/game-graybox.html

Texture Animations

```
Environment.prototype.update = function()
{
        if (this.textureSky)
        {
                this.sky.material.map.offset.x += 0.00005;
        }

        if (this.app.running)
        {
                var now = Date.now();
                var deltat = now - this.curTime;
                this.curTime = now;

                dist = -deltat / 1000 * this.app.player.speed;
                this.road.material.map.offset.y += (dist*Environment.ANIMATE_ROAD_FACTOR);
        }

        Sim.Object.prototype.update.call(this);
}
```

Code
Chapter 8/game-graybox.html

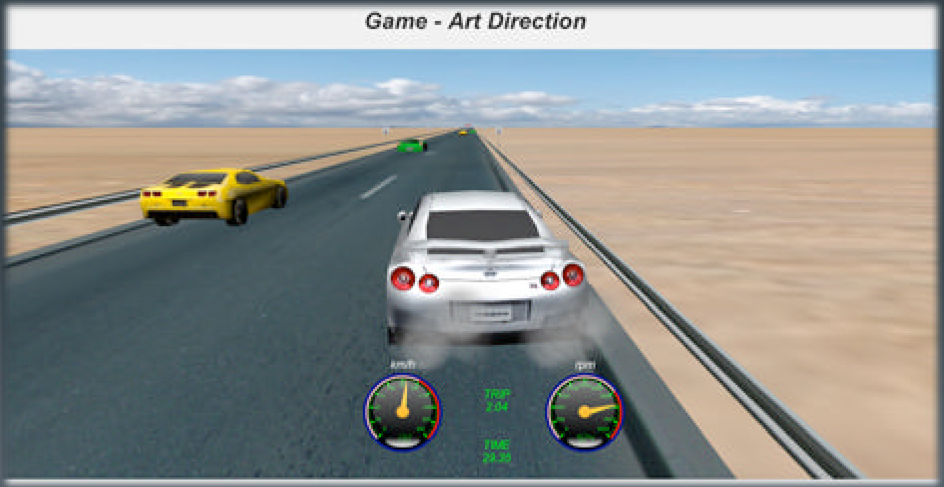
Collision Detection

```
RacingGame.prototype.testCollision = function()
{
            var playerpos = this.player.object3D.position;

            if (playerpos.x > (Environment.ROAD_WIDTH / 2 - (Car.CAR_WIDTH/2)))
            {
                        this.player.bounce();
                        this.player.object3D.position.x -= 1;
            }

            if (playerpos.x < -(Environment.ROAD_WIDTH / 2 - (Car.CAR_WIDTH/2)))
            {
                        this.player.bounce();
                        this.player.object3D.position.x += 1;
            }

            var i, len = this.cars.length;
            for (i = 0; i < len; i++)
            {
                        var carpos = this.cars[i].object3D.position;
                        var dist = playerpos.distanceTo(carpos);
                        if (dist < RacingGame.COLLIDE_RADIUS)
                        {
                                    this.crash(this.cars[i]);
                                    break;
                        }
            }
}
```

Code
Chapter 8/game-graybox.html

Art Direction

The Art Direction Study



Code

Chapter 8/game-art-direction.html

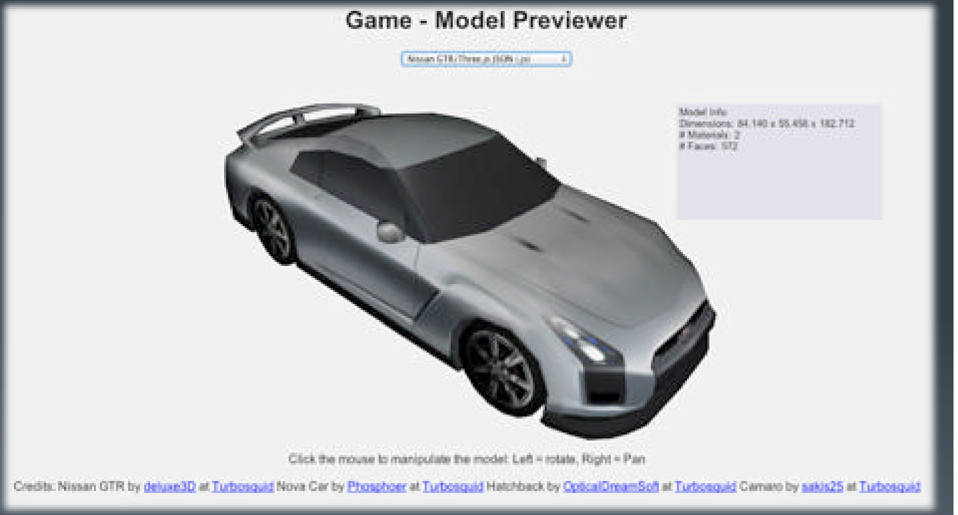
The Heads-Up Display

- WebGL's 2D/3D integration in action
- Open source speedometer widget by Marcello Barnaba
  - Fully configurable
  - All HTML5, CSS, JavaScript!
  - https://github.com/vjt/canvas-speedometer

Code
Chapter 8/game-art-direction.html

The Model Previewer

- Models from Turbosquid
- Previewer to get sizes, inspect normals, get model stats
- Overall, Tools and Exporters Still Primitive



Game - Model Previewer

Model Info:
Dimensions: 84.140 x 55.456 x 182.712
Materials: 2
Faces: 572

Click the mouse to manipulate the model: Left = rotate, Right = Pan

Credits: Nissan GTR by deluxe3D at Turbosquid Nova Car by Phosphoer at Turbosquid Hatchback by OpticalDreamSoft at Turbosquid Camaro by sakis25 at Turbosquid

Code
Chapter 8/game-model-viewer.html

Building a Particle System

The Particle System Lab

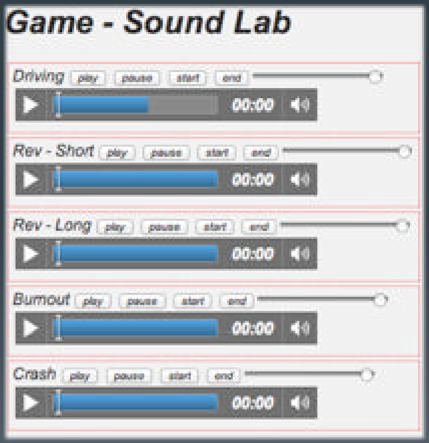
- Three.js has a basic built-in particle system
- But It's very basic: No emitters or physics models
- You have to animate it all yourself

Code
Chapter 8/game-particles.html

Adding Sound

- Use new `<audio>` Element
- Fairly well supported in browsers
- Other APIs (Moz, Chrome) not supported uniformly

The Sound Lab



Code
Chapter 8/game-sounds.html

Making It Robust

- Detecting WebGL support in the browser

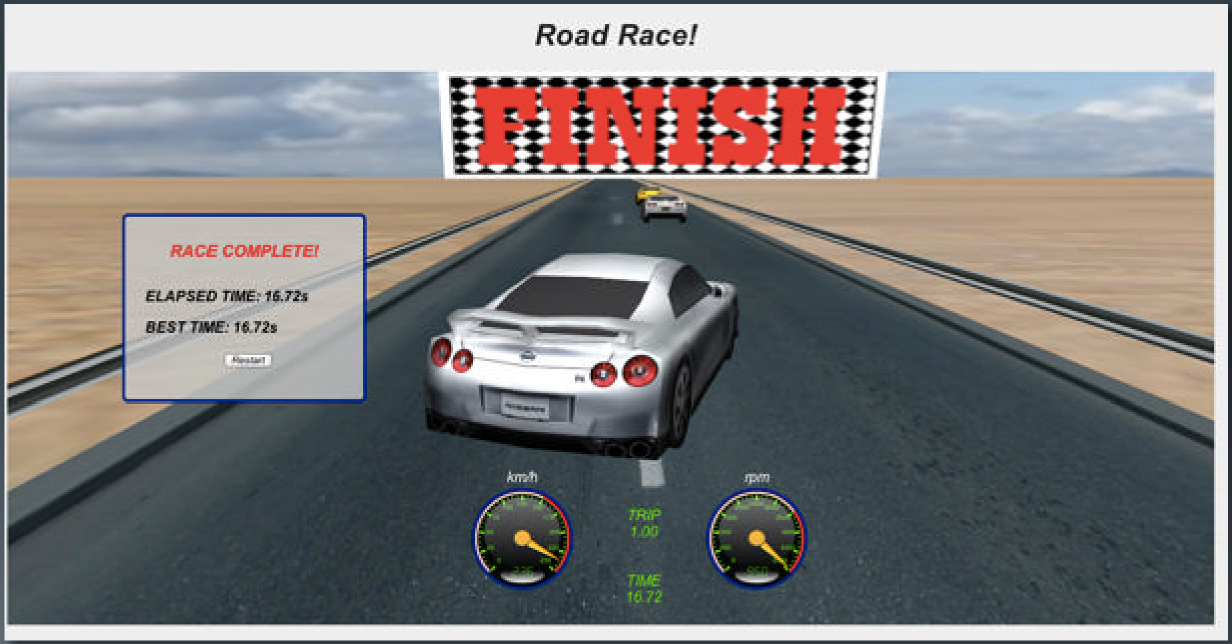
```
var canvas = document.createElement("canvas");

var gl = null;
var msg = "Your browser does not support WebGL.";
try
{
  gl = canvas.getContext("experimental-webgl");
}
catch (e)
{
  msg = "Error creating WebGL Context!: " + e.toString();
}
if (!gl)
{
  throw new Error(msg);
}
```

Making It Robust

- Detecting a lost context

```
RacingGame.prototype.handleContextLost = function(e)
{
          this.restart();
}

RacingGame.prototype.addContextListener = function()
{
          var that = this;

          this.renderer.domElement.addEventListener("webglcontextlost",
                              function(e) {
                                        that.handleContextLost(e);
                                        },
                              false);
}
```

Putting It All Together



Code
Chapter 8/game.html

The Bottom Line

- WebGL Is really solid
  - OpenGL ES under the hood (It's what's running on your phone and tablet)
  - Huge development, testing and conformance effort by browser writers
  - Strong standards group maintaining it (www.khronos.org)
- A few enhancements will help
  - Transferables, Built-in Matrices
- Production capability is still very crude
  - Tools and frameworks are young and evolving
  - Export from pro tools lacking
- **The real issues facing game developers**
  - **JavaScript runtime (e.g. garbage collection)**
  - **Device input**
  - **Audio and video API chaos**
  - **Formats and delivery - streaming, compression, binary**

Where To Go From Here

- WebGL development – not easy; but not rocket science
- Small feature set with huge capabilities
  - This talk is the tip of a deep iceberg
- WebGL has moved from the infant to the toddler stage
  - The standard is well thought out and well vetted
  - Implementations are fast, robust and work identically across browsers
  - Tools are still rough
- Three.js is one of many frameworks
  - Look around, or build your own
- The possibilities are limitless!

Resources

- WebGL Specification
  - http://www.khronos.org/webgl/
- Learning Sites
  - http://learningwebgl.com/blog/
  - http://learningthreejs.com/
- News and Info
  - http://www.reddit.com/r/threejs
  - http://www.webgl.com/
- Engines
  - https://github.com/mrdoob/three.js/
  - http://www.glge.org/
  - http://www.scenejs.org/
  - http://www.cubicvr.org/
- Demos
  - http://www.chromeexperiments.com/webgl
  - https://developer.mozilla.org/en-US/demos/tag/tech:webgl/
  - http://chrysaora.com/
- Tools
  - http://www.ambiera.com/copperlicht/index.html
  - http://www.sunglass.io/

Thanks!

The Code
https://github.com/tparisi/WebGLBook/

Contact Information
tparisi@gmail.com
Skype: auradeluxe
http://twitter.com/auradeluxe
http://www.tonyparisi.com/

Get the book!
eBook/Early Release: May 22, 2012
Print version: July 2012
http://shop.oreilly.com/product/0636920024729.do
Discount Code: WEBGLPRE