WebGL: a new standard for developing 3D applications

Elisabete Thomaselli Nogueira
Rio de Janeiro Federal University
COC/COPPE/UFRJ
elisabetetn2004@gmail.com

Abstract

This research presents an exploratory study on Web Graphics Library (WebGL) and its benefits in 3D software application development, applied to different areas. The results showed that data are displayed, with neither software dependent platform nor installation needs, intent to be a great alternative to 3D applications such as “WebGL playground”, a development environment to WebGL and the interactive movie “3 Dreams of Black”.

Keywords: WebGL, Computer Graphics, Augmented Reality

Introduction

WebGL defines a JavaScript binding to OpenGL@ES (Open Graphics Language) (MUNSHI, A. at al, 2008), which allows rich 3D graphics use without plug-ins. The Canvas element is used as presentation level in HTML5, in Firefox, Chrome, Opera or Safari implementations (LEMKE, C., 2011 e PEREIRA, A. P., 2010).

The WebGL workgroup joins Apple, Google, Mozilla and Opera browsers makers. WebGL can be used on desktop computers, mobile and embedded computer platforms, with support to OpenGL (SHREINER, D. 2010) or OpenGL@ES 2.0.

The WebGL 1.0 specification was announced by Khronos Group for the first time in 2009, March, in the Game Developers Conference, in San Francisco, CA.

Khronos Group organized a working group to the WebCL specification to describe JavaScript interfaces to OpenCL (Open Computing Language). By this mean, Web
applications will be capable to do better use of the computer’ architectures, then, once developers can codify WEB applications that takes advantage of GPU and multi-core processors as well. According to Khronos Group, WebCL can be fit to image and video processing and advanced physics.

**Browsers’ version**

WebGL is supported by Chrome 9.0 e Firefox 4.0, among others (WEBGL, 2012).

**The most common routines**

The scripts bellows are used in most of WebGL applications. The scripts are:

- **glMatrix-0.9.5.min.js** → Designed to handle matrix operations and vector operations in high performance, such as: subtraction, transposition, calculate the determinant, vector normalization, translation, rotation and scale matrix, among others.

- **webgl-utils.js** → It handles basic treatments for common used routines, as, for example, create a WebGL context, return error messages from html, animation calls in the browser, among others.

**Installation Test**

Once you have a browser installed, you should be able to have your WebGL contents displayed correctly. A simple sanity check can be done by use of the WebGL Report (REPORT, 2012), which presents a browser checklist of functionalities (See Picture 1). In case of error return, in other words, if your installed browser does not support WebGL, consult troubleshooting guide also (VUKIČEVIĆ, V. at al., 2012).
Lessons

Some lessons are available at internet. Among them, take the site Learning WebGL (LEARNING, 2012), where there are posted some varied examples, from displaying text information and particles to programmed animations. The source code to this simple sequence of lessons and examples listed are catalogued on github repository (THOMAS, G. 2012), using concepts and commands originated from openGL and, in many cases, based on well-known similar examples made in openGL.
Basic Tools: Editors, Trace, Exporters and others

We make available below a first listing with Editors (PSPad and WebGL Playground), Trace (WebGL Trace), Converters (From Blender to JS), Frameworks (PhiloGL and GLGE), and Libraries (SpiderGL), among others, that may support development with WebGL.

Editors

PSPad

The PSPad is a very good free editor to JavaScript and html.


The PSPad editor accept files from types C++, Cobol, MS-Dos batch, CSS, Fortran, FoxPro, HTML, XHTML, INI, Inno Setup, Java, JavaScript, KixStart, Object Pascal, Perl, PHP, Python, RSS, SQL, TCL/TK, Unix ShellScript, VBScript, Visual Basic, X86 assembler and more.

WebGL playground

The WebGL playground is a free tool that helps people to write the WebGL script and see the visualization results on the same page, in an organized way and with syntax highlighted (CONRAD, D., 2012). The editor allows us to work with JavaScript code and GLSL vertex/fragment shader (if there it be any) on the same page in a convenient way.

On URL (http://webglplayground.net/gallery), there are some very interesting examples, from fractals (http://webglplayground.net/?gallery=lsystem-tree-fractal) and fluids to organic networks, which can be used as template ones.
WebGL-Trace

It is a trace utility for WebGL development. Webgl-trace (PALEVICH, J., 2010) is based on the webgl-debug utility and gives additional information about texture objects, recognize and handle array overlays.

Exporters

From Blender to JS

To export objects from Blender (ROOSENDAAL, T., 2005) to WebGL JavaScript, we can use the webGLExport.py plugin (IPPEL, D. V., 2010). Using webGLExport.py is possible to export: all selected meshes or all meshes in the scene, vertexes, normals, UV coordinates, vertex colors and simple frame animation.

Actually, it is possible to export files from Blender version 2.44, using webGLExport.py plug in. Considering that your Windows has native support to WebGL, GLGE and SceneJS, it is necessary to have the blender software and python 2.5 installed and model in .blend format, also.

Frameworks

PhiloGL

PhiloGL (BELMONT, N. G., 2011) is a WebGL Framework for Data Visualization, creative coding and game development. This high level API reduces the WebGL programming effort. Some examples from URL (www.learningwebgl.com) were converted to it and are available in the demo directory, inside the site product itself.

A quite interesting example of PhiloGL code is the tutorial (METAMOLECULAR, 2012), which shows a benzene molecule visualization in details.
GLGE – WEBGL for lazy

GLGE (GLGE, 2012) is a JavaScript library that intend to simplify the use of WebGL; as we have already known, a native JavaScript API to browsers, providing direct access to openGL@ES2 and allowing hardware accelerating in applications 2D or 3D, with no need of plug ins download.

An application example of use of GLGE (and JSTweener (TATENO, Y., 2012) and webGLExport.py exporter also) is “Unused WebGL Presentation made with WebGL”, by Rozeng (SOLYGA, P., 2010), tested on Firefox e Mozilla Nightly browsers only. See Picture 2. To change to another slide, press 1 to 6 in the keyboard.

![Picture 2: Presentation Scenes.](image)

Libraries

SpiderGL

It is a 3D graphical library to render WebGL in real time (DI BENEDETTIO, M., 2010), which has some components, as described above:

MATH : Math routines, low/high level linear algebra functions and classes.
SPACE : Geometric structures and space-related classes and algorithms.
GL : Low/high level WebGL resources access, rendering.
MESH : Editable and renderable meshes, importers, rendering.
UI : User interface event handling, interactors.
DOM : JS/HTML utilities, asynchronous content loading.

DOWNLOAD URL (http://spidergl.org/).
The Picture 3 shows an example of how to use HDR textures using PNGHDR file format, encoding extra information in the alpha channel of a PNG file. In Picture 4 (a), we can see a Ray-Casting demonstration and in Picture 4 (b) we have a shadow mapping instead.

**Picture 3: spidergl: hdr texture (Banterle, F.).**

**Picture 4: (a) Ray-Casting with BlockMaps, (b) Shadow Mapping.**

Much more information about SpiderGL components can be get in the paper “SpiderGL : A JavaScript 3D Graphics Library for Next-Generation” accepted on
“Web3D 2010 Conference” (BENEDETTO, M. et al., 2010). A generic architecture schema is available in Picture 5.

![SpiderGL architecture diagram](image1)

**Picture 5: SpiderGL architecture diagram.**

Other interesting SpiderGL use example is related to molecular structure visualization (CALLIERI, M. et al., 2010). In this example, there was used the PDB importing file format, considered a very common biology file format to describe molecular structures, in a few code lines. In this context, it was concluded the advantages of construct 3D interactive visualization schemas to scientific data that were produced by researches in cellular and molecular biology, in WEB platform. See Picture 6.

![SpiderGL Molecular visualization](image2)

**Picture 6: SpiderGL Molecular visualization.**
Examples: Simple ones, More complex one and Academic papers.

Simple examples

A drawed triangle in WebGL window:


Texture and movement sample: A cube with only one texture and a fixe animation – (http://learningwebgl.com/blog/?p=507).

The WebGL is able to use up to 32 textures during any function call, similar to gl.drawElements function, that organize these textures from TEXTURE0 to TEXTURE31. In this example from lesson 5, only one texture is used (TEXTURE0). Observe: The image height and width have to be numbers that are multiple from 2 (number of pixels).

An’ animation done by a regular call of tick function, which renew the animation state; redraw the scene and guarantee that tick call will be called again in a stipulated time (the triangle has movements of 90° /s and the square in angle of 75° /s) – (http://learningwebgl.com/blog/?p=370).

Tunnel: In WebGL export plug in webGLExport.py’ author site, it is possible to visualize a very interesting example named “Using WebGL & GLSL Shaders to Create a Tunnel Effect”, which has texture uses, and animation also, and gives us a tunnel illusion’ effect – (http://www.rozengain.com/blog/2010/08/10/using-webgl-glsl-shaders-to-create-a-tunnel-effect/).
More complex examples

The following examples are pretty interesting for the element composition presented.

HelloRacer: High-end interactive car racing simulation. Use the keyboard arrows to move the car. This simulation uses Three.js resources.

It can be access in URL (http://helloracer.com/webgl/).

Music: Lose yourself in a Music Experience by @EllieGoulding’s with Lights, interactivity and colorful music video using WebGL. Keep the Button pressed to fly faster.

It can be access in URL (http://lights.elliegoulding.com/).

Particles

Picture 7 example implements blaze, smoke, text use, and so on.

It can be access in URL


This Chromium demonstration, a free software project from Google Chrome, uses the open-source API JavaScript O3D to create interactive graphic application to run in a browser - games, ads, 3D model visualizing, product demonstrations and virtual worlds.
O3D (TAVARES, G. 2010) spread the client side from a WEB application by given more functionality in the system level, content level and code level, as following:

System: O3D API provides a plugin for browsers, adding graphical functionalities for browsers in Windows, Macintosh and Linux platforms.

Content: HTML forms, image and video files. It allows us to create exporters and importers to any 3D content. O3D has an example named COLLADA Converter, which can be used to import files from COLLADA format, a standard to 3D, supported in known applications as SketchUp, 3ds Max and Maya.

Code: O3D spreads JavaScript applications with a 3D graphic Application Program Interface (API). It uses standard JavaScript event processing and callback methods.

Concerned that JavaScript would be slow to drive a low-level API like OpenGL and that we were also cognizant of the lack of installed OpenGL drivers on many Windows computers, O3D scene graph would yield better results, at the beginning. Although, the O3D plug in development has been stopped, since JavaScript has becoming better and many functions, which were treated as O3DJS before, has emerging in the WebGL standard code.
Academic papers from IEEE

Academic studies are testing WebGL different use aspects. They are demonstrating WebGL feasibility and performance in different contexts. There are some of them, as bellow:

In the “Extending Web Applications with 3D Features” paper (LONGO, M. e VAIRA, L., 2011), the possibility to add 3D interface objects to existing web applications was explored. The proposed approach is reusable, because it is independent from the specific application, it adopts the same technologies and software components normally used in three layer (or n-layers) applications and it has been validated not even in proprietary tools like MS Visual Studio and MS SQL Server but also in open tools and languages, like PHP and MySQL. However, much work must be done to promote html5 and WebGL from the role of enabling technology to the role of solid foundation for enterprise level applications.

A browser-based Multiplayer Online Game system (MOG), which require no explicit installation and is able to achieve cross-platform easily, are getting more and more popular. The “A WebGL-Based Method for Visualization of Intelligent Grid” paper (WEIGANG, Z. at al., 2011) shows a MOG based on WebGL and WebSocket, implementing such a framework for browserbased multiplayer online games and studied its performance and feasibility. As analytical result showed, Three.js 3D Engine and jWebSocket based MOG can easily support the interaction of a small group of users, which seem to be very promising.

On the other hand, the “A WebGL-Based Method for Visualization of Intelligent Grid” (WEIGANG, Z. at al., 2011) paper introduces a visualization system model, which is based on X3DOM framework that base on WebGL to display components and their run-time states of intelligent grid.

Last, the “WebGL implementation in WebKit based web browser on Android Platform” (GOLUBOVIC, D. at al., 2011) paper presents one solution for WebGL implementation on ARMv7 set-top box with Android Gingerbread as target platform.
Application Development (JsARToolKit, Video Clip, Interactive Movie and GOOGLE MAPS)

From now on, we will show a few more complex examples, as, for example, interactive movies and development codes resources (as frameworks or high level interface languages).

A complete WebGL 1.0 documentation, the current specification, besides some implementations and demonstrations can be finding in the public wiki (WebGL, 2012). This reference can contribute significantly to the Knowledge of other applications cited inside these papers on the whole.

**JsARToolKit –**

There is an Augmented Reality (AR) demonstration, made in WebGL (IMARIEHI, 2012) available in Mozilla web page, which can be observed in Picture 8. On this demonstration, the positioning is realized throughout video, and not by webcams, what seems to be the most usual way to do that. Even in this case, the AR system is based on AR library to JavaScript (IMARIEHI, 2012). Additionally, there is a mobile platform, in beta test version (KAWASAKI, Y., 2012). In this case, the JavaScript seems to do a link between ARToolkit (RA for Flash) and WebGL. At all, the Mozilla web page has a plenty of interesting demonstration examples (URL https://developer.mozilla.org/pt-BR/demos/ ).
Video Clip

A video clip with videos and animated 3D models to a music were created in no-comply (ROUGET, P., 2012), using WebGL and AudioAPI technology either.

This video clip uses the WebM format file (WebM, 2012), an audio and video format file with authority/intellectual property rights without costs and video compressing to be used in video HTML5. This project development was paid by Google.

Interactive Movie

"3 Dreams of Black" is an interactive movie that wants to show the creative possibilities to WebGL. In Picture 9, it can be found many scenes and content views from the "3 Dreams of Black" movie, with videos, new created scenes (Picture 10), sound tracks and mouse interactions.

On “3 Dreams of Black” website (MICK, C., 2012) it can be found 8 (eight) demonstrations cited by development group of Google Chrome also, basic code (Apache license 2.0) and links to WebGL resources.
GOOGLE MAPS

Nowadays, Google Maps published a Google Maps version in WebGL (PARKER, E., 2012). In this case, users are invited to test MapsGL, Beta technology from Google Maps, activated by WebGL. See Pictures 11 and 12. More than over, to access 3D elements, there must be a Google Earth plug in installed (Picture 13).
Do yourself: scientific development

On Google Summer

In 2011 edition, Google Summer Code proposes a WebGL implementation to Volume Rendering. They suggested that an initial ray-cast volume rendering implementation in WebGL could be done. The students would explore each
technology and previous work existing using shader routines, which emulates behaviors implemented in C++ VTK libraries and measure the results and WebGL implementation precision comparing with exist C++ ones. They expected a JavaScript library construction to get 3D images as input and produced an interactive volume rendering interactive on browser as output (SCHROEDER, W. at al., 2010). This project had Jeff Baumes as mentor (jeff.baumes at kitware dot com).

The 2012 event is opened to proposals yet (GOOGLE, 2012).

In Linux

When telling about Linux Operational System, WebGL seems to work properly on Ubuntu, Chrome 10 browser, since drives were installed correctly (UBUNTU, 2012).

Conclusion

This survey intent to be a first look at of one of the newest 3D software technologies available nowadays. Some useful libraries, converters, editors and frameworks were presented. Details about some choices of software and hardware and final definition of the platform were discussed. The results obtained with these 3D experiments in WebGL and their gadgets were very satisfactory, considering the main objective of the project that was to investigate the WebGL features and user hopes. The ease of use of the 3D software based in WebGL must be highlighted also. The 3D limits of the use of WebGL may be only related to the difference among browsers and the newness aspect of this technology, what seems to be temporary.

References


Visited URLs


About the Author

Elisabete Thomaselli Nogueira