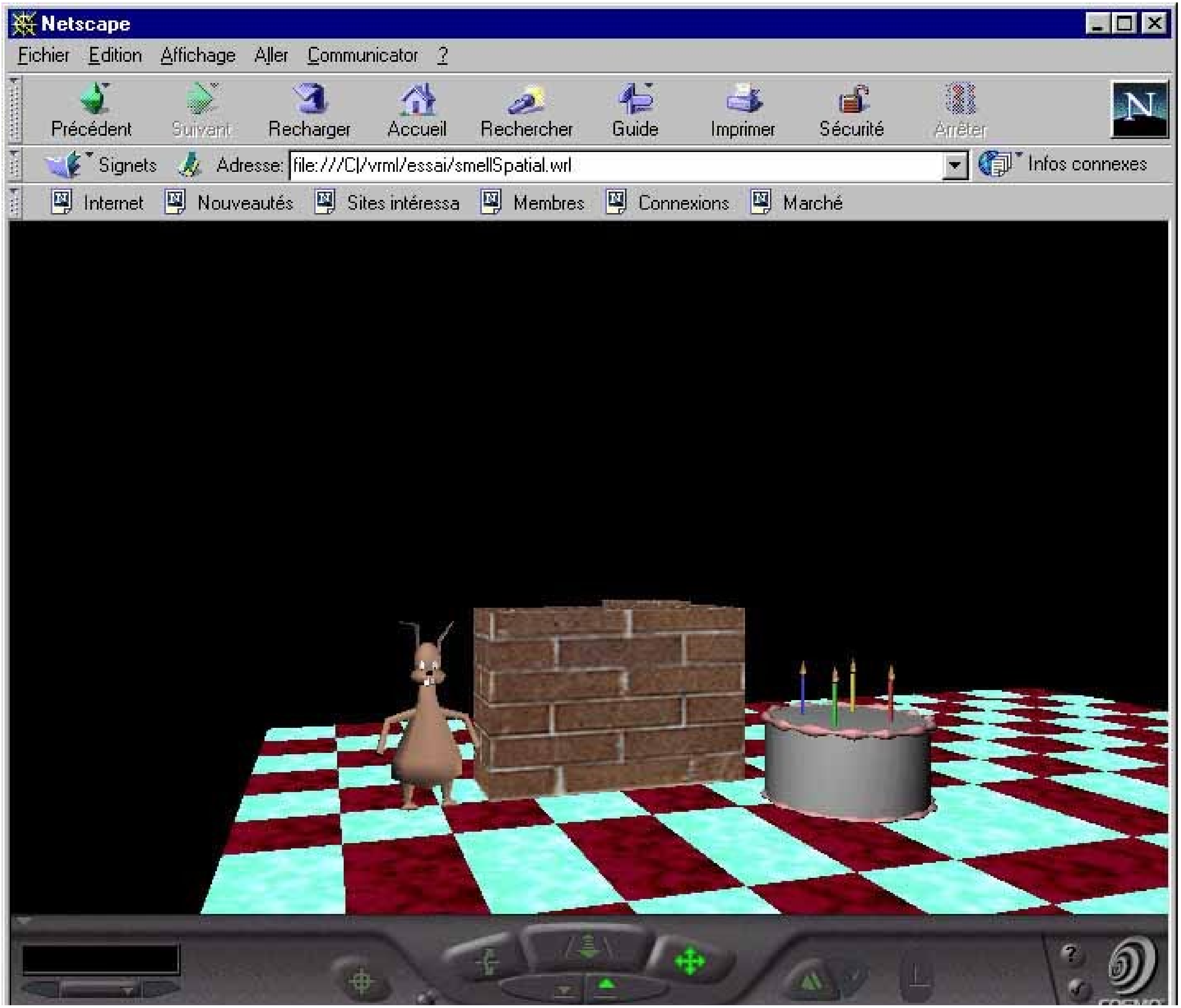
webGL

Anton Gerdelan
<gerdela@scss.tcd.ie>
antongerdelan.net
Trinity College Dublin

bkrd

- People have wanted web virtual ~ for ages
- VRML (1994)
- Second Life
- Early engine browser plug-ins
  - OGRE (~2005)
  - Unity 3d


Netscape
Fichier Edition Affichage Aller Communicator ?
Précédent Suivant Recharger Accueil Rechercher Guide Imprimer Sécurité Arrêter
Signets Adresse: file:///C|/vrml/essai/smellSpatial.wrl
Infos connexes
Internet Nouveautés Sites intéressa Membres Connexions Marché

source: http://pauillac.inria.fr/~codognet/vrml/creatures.html

what went wrong?

- slow download/parsing
- slow rendering
  - In-browser execution
  - antiquated 3d tech. / no shaders / clunky interface
- download proprietary plug-in
- limited to particular browsers or platforms

since

- Mobile development and Opengl ES
  - Cut-down [reliable] version of OpenGL
  - Catered to web-developers; `n = 100n`
- Microsoft failed {SL,DX Mob/web}
- Flash died
- Everything has a GPU
- HTML5
- Mozilla `<canvas>` experiments hack OpenGL in

webgl

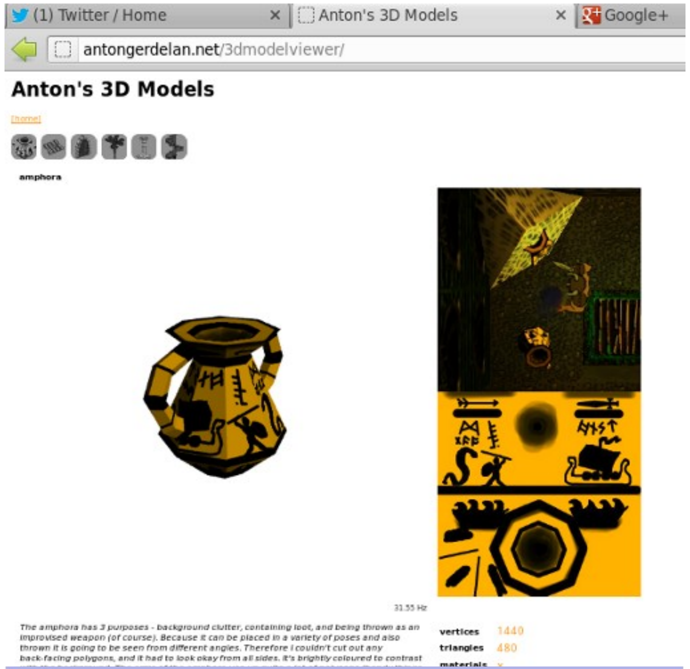
- Access the GPU and write shaders
- **Browser is the platform***
- Interface based on cut-down OpenGL ES
- *Interface must be implemented by vendor of browser
- Compare this to traditional limitations of
  - Direct3D
  - OpenGL
- Security concerns and big corp. hold-outs ($$$$$$).

wha

- games
- 2d and 3d vector graphics
- interactive visual experience
- extremely fancy advertisements
- virtual reality worlds*

*supporting technology pending

xp


(1) Twitter / Home
Anton's 3D Models
Google+
antongerdelan.net/3dmodelviewer/

Anton's 3D Models

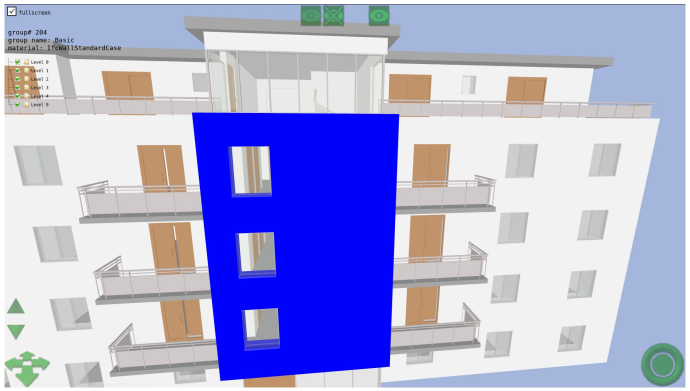
[home]

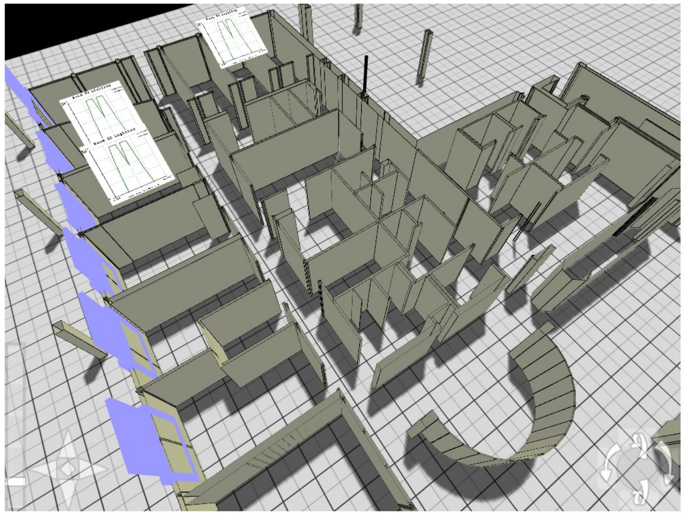
amphora

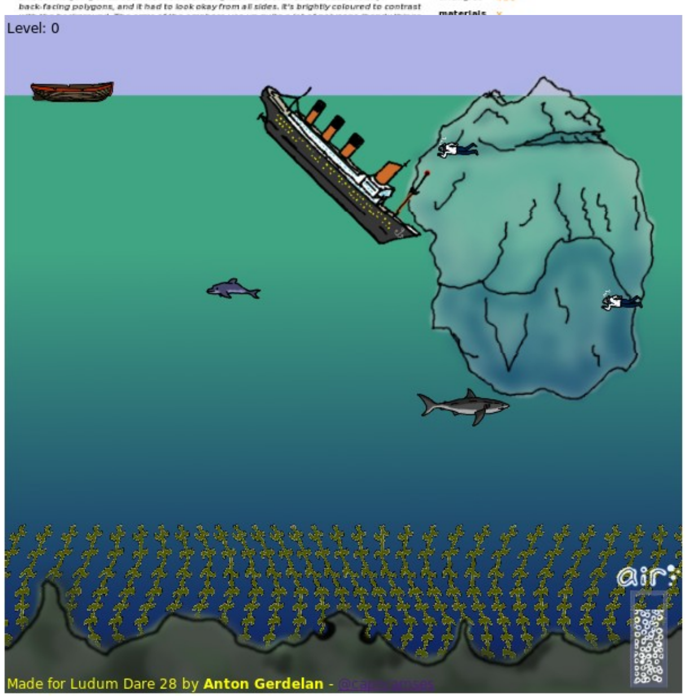
31.55 Hz

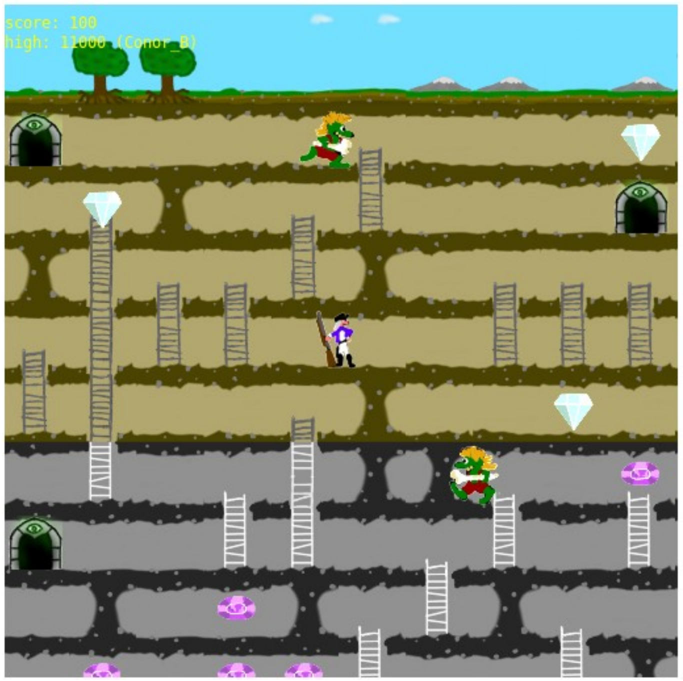
The amphora has 3 purposes - background clutter, containing loot, and being thrown as an improvised weapon (of course). Because it can be placed in a variety of poses and also thrown it is going to be seen from different angles. Therefore I couldn't cut out any back-facing polygons, and it had to look okay from all sides. It's brightly coloured to contrast

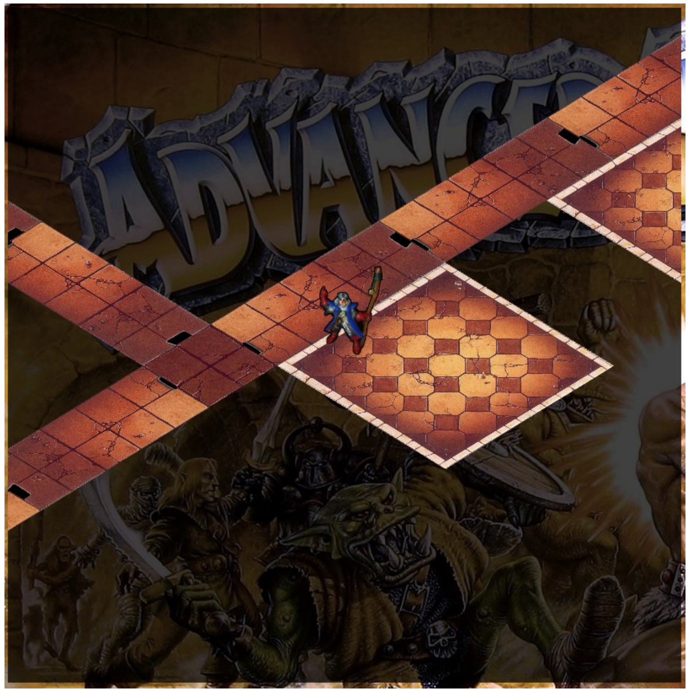
vertices	1440
triangles	480
materials	x











how

- Default interface through JavaScript (hmm…)
  - 90% the same as modern OpenGL with C or OpenGL ES
- Write **shaders** in GLSL (exactly the same as ES)
- But…
  - three.js {engine, web-designers, ~GLUT}
  - Dart (OO, compiles to fast JS)
  - emscripten **transcompiler** to JS

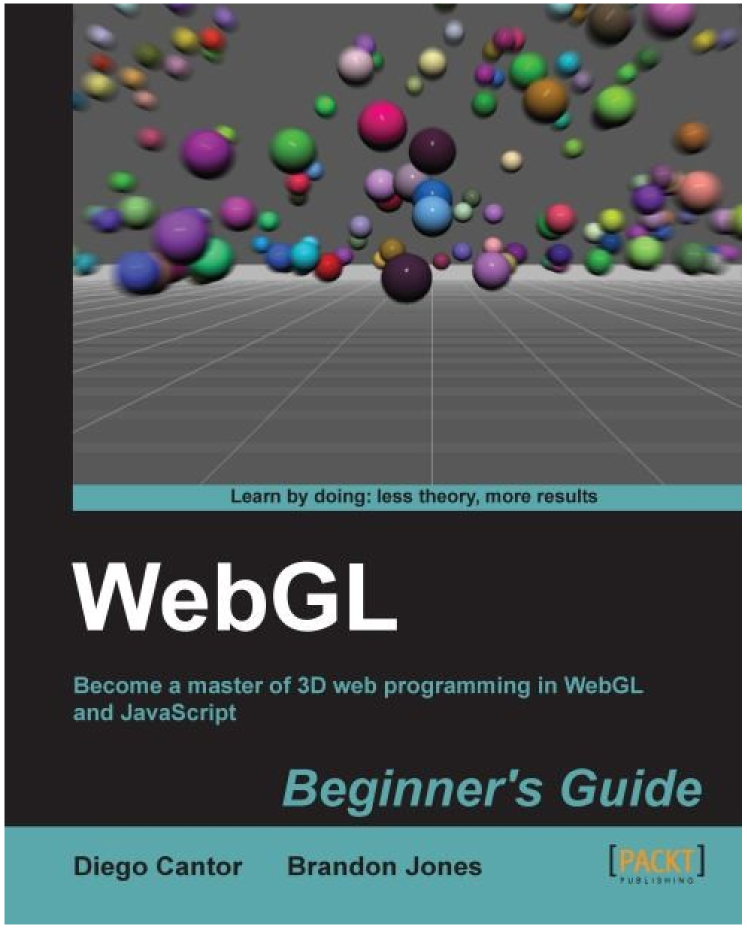
why

- v quick dev vs compiled
- build once and run on <u>everything</u> (incl mobile) (ask about OpenGL dev)
- no plugins
- combine w websites and web interfaces (ask about Qt + 3d)
- great for portfolios/sharing/tinkering and impressing EU project reviewers

whr

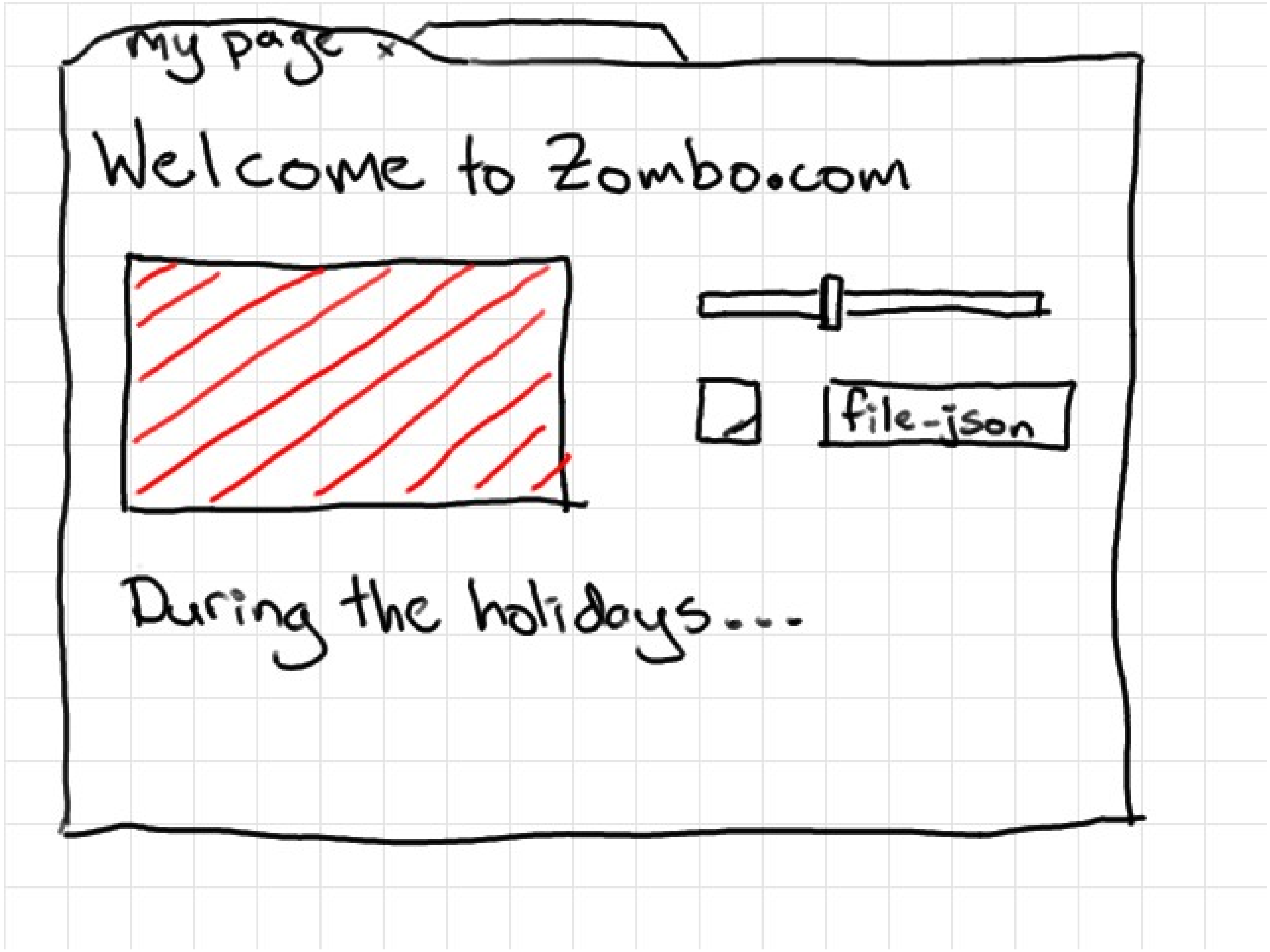
- basic webgl and JavaScript →
  - http://learningwebgl.com/
- three.js – website has tuts.
  - And lots of really nice demos/experiments
- Dart – website has tuts.

- codeflow.org – a great blog
- Udacity – free/paid online 3d graphics course
  https://www.udacity.com/course/cs291



Learn by doing: less theory, more results
WebGL
Become a master of 3D web programming in WebGL and JavaScript
Beginner's Guide
Diego Cantor
Brandon Jones
[PACKT] PUBLISHING

ex



ex

```
<canvas id="mycanvas">
</canvas>
```

Next get “WebGL 1.0 Quick Reference Card” from the Khronos Group as primary reference
https://www.khronos.org/files/webgl/webgl-reference-card-1_0.pdf

ex – try now?

```
<!DOCTYPE html>
<html><body>
<canvas id="canvas"></canvas>

<script type="text/javascript">
var canvas = document.getElementById ("canvas");
var gl = canvas.getContext ("webgl");
gl.clearColor (1.0, 0.0, 0.0, 1.0);
gl.clear (gl.COLOR_BUFFER_BIT);
</script>

</body></html>
```

Ex – 3d points that make a triangle

- Triangle has how many 3d points?
- Screen area is in range of -1.0 to 1.0 on x,y,z axes
- Make a 1d array containing x,y,z,x,y,z,x,y,z

```
var pts = [
-0.5, -0.5, 0.0,
0.0, 0.5, 0.0,
0.5, -0.5, 0.0
];
```

Ex – copy points into GPU memory buffer

```
var vbo = gl.createBuffer ();
gl.bindBuffer (gl.ARRAY_BUFFER, vbo);
gl.bufferData (gl.ARRAY_BUFFER, new
Float32Array (pts), gl.STATIC_DRAW);
```

- “vertex buffer object” (VBO)
- old-fashioned “binding” conventions
- see reference card for details

Ex - shaders

- Rendering anything requires a “shader program” which defines a style of rendering.
- Two parts
  - Vertex shader (how to position each vertex point on screen)
  - Fragment shader (how to colour each pixel-sized piece of triangle)

Ex – vertex shader

- Write/read a little GLSL programme into a JS string.
- Looks like C.

Input a single xyz point

```
var vs_str =
"attribute vec3 vp;" +
"void main () {" +
"  gl_Position = vec4 (vp, 1.0);" +
"}";
```

Output an xyzw point i.e.
“put straight on screen”

Ex-fragment shader

- Pretty much the same style
- Output is a colour in RGBA
- **Q. What colour will the triangle be?**

```
var fs_str =
"precision mediump float;" +
"void main () {" +
"  gl_FragColor = vec4 (0.0, 0.0, 1.0, 1.0);" +
"}";
```

red, green, blue, alpha

Ex – compile shaders, link together

```
var vs = gl.createShader (gl.VERTEX_SHADER);
var fs = gl.createShader (gl.FRAGMENT_SHADER);
gl.shaderSource (vs, vs_str);
gl.shaderSource (fs, fs_str);
gl.compileShader (vs);
gl.compileShader (fs);
var sp = gl.createProgram ();
gl.attachShader (sp, vs);
gl.attachShader (sp, fs);
gl.linkProgram (sp);
```

- Tedious busy-work
- Compiles each mini-program, links together so it will run <u>on the GPU</u>

Ex – use program, draw vertex buffer

```
gl.useProgram (sp);
gl.bindBuffer (gl.ARRAY_BUFFER, vbo);
gl.vertexAttribPointer (0, 3, gl.FLOAT,
false, 0, 0);
gl.enableVertexAttribArray (0);
gl.drawArrays (gl.TRIANGLES, 0, 3);
```

- Enable shader and buffer of points
- Describe data format (every 3 floats is a variable)
- Draw 3 points from buffer in triangles mode

ex

- Should work
- 49 lines of html + js + glsl
- Interface a bit tedious
  - hide away in a “utils” file or
  - use a framework like three.js
- Can now
  - Draw more triangles (from a mesh file)
  - Make fancier shaders
  - Add interaction, animation, sounds, etc.

skills

- JavaScript and HTML
- Linear algebra (vectors, matrices)
- GLSL shaders (not very hard, but strange)
- (Can skip / hide from some stuff by using three.js)
- Eye for good visual design (or actual theory)
  - Colours
  - Spatial/depth
  - User interaction

pers

- Is this an easy/fun way to get into 3d? = **yes**
- Is this a quick/powerful game jam platform? = **yes**
- Issues?
  - JavaScript bugs
  - Not quite cutting edge gfx
  - All your code is visible
  - Compat/cross mostly solved
  - Wide range of hardware - `min()`

pers

- Problems to solve
  - Multi-player / multi-user.
  - Commercialising
  - Streaming/handling large files (meshes/textures)
- New stuff
  - Touchscreens
  - Fullscreen
  - Websockets
  - Game input handler (joysticks/gamepads)

warez

- I have loads of stuff on github https://github.com/capnramses
- And my games are playable from http://antongerdelan.net/games/
- And my email: gerdela@scss.tcd.ie
- And I live in F.30-something (one of the big bull-pens) in the top-floor of O'Reilly