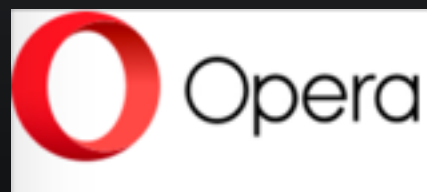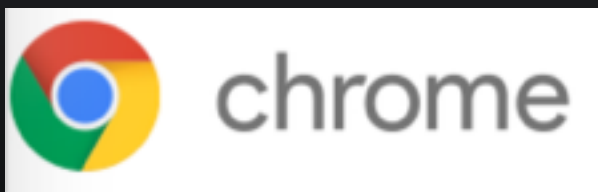# Introduction to WebGL

COMP 531 presentation

Jing Guo

11/29/2016

WebGL = Web Graphics Library

- Render 2D/3D graphics in the browser
  - ✦ Provide Javascript API based on OpenGL ES 2.0
  - ✦ <u>An example</u>
- No plug-in required, access to GPU
  - ✦ Powerful, scalable, flexible
  - ✦ Render to `<canvas>` element in HTML
- Support by many browsers

Implementation of WebGL

```
1   <body onload="start()">
2     <canvas id="glcanvas" width="640" height="480">
3       Your browser doesn't appear to support the
4       <code>&lt;canvas&gt;</code> element.
5     </canvas>
6   </body>
```

- Initialization
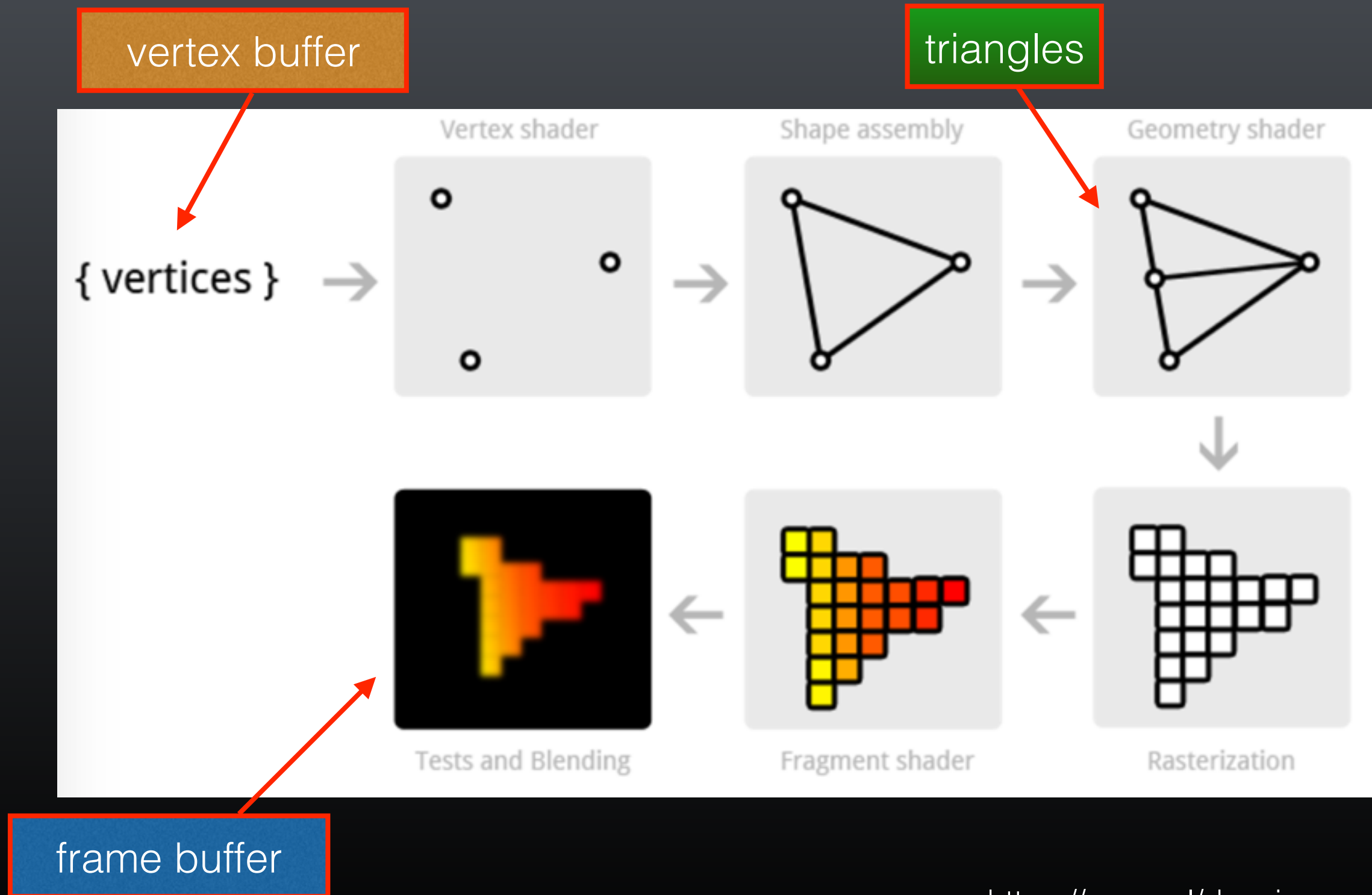 - ✦ Create <canvas> tag
  - ✦ Establish reference

```
1   var gl; // A global variable for the WebGL context
2
3   function start() {
4     var canvas = document.getElementById("glcanvas");
5
6     // Initialize the GL context
7     gl = initWebGL(canvas);
8
9     // Only continue if WebGL is available and working
10    if (!gl) {
11      return;
12    }
13
14    // Set clear color to black, fully opaque
15    gl.clearColor(0.0, 0.0, 0.0, 1.0);
16    // Enable depth testing
17    gl.enable(gl.DEPTH_TEST);
18    // Near things obscure far things
19    gl.depthFunc(gl.LEQUAL);
20    // Clear the color as well as the depth buffer.
21    gl.clear(gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT);
22  }
```

https://developer.mozilla.org/en-US/docs/Web/API/WebGL_API]

Initialization of Shaders

- Vertex shader

  ✦ Compute attributes of vertices: projected position, color, texture …

- fragment shader

  ✦ Fragment: individual pixel

  ✦ Compute attributes of fragment

```javascript
function initShaders() {
  var fragmentShader = getShader(gl, "shader-fs");
  var vertexShader = getShader(gl, "shader-vs");

  // Create the shader program

  shaderProgram = gl.createProgram();
  gl.attachShader(shaderProgram, vertexShader);
  gl.attachShader(shaderProgram, fragmentShader);
  gl.linkProgram(shaderProgram);

  // If creating the shader program failed, alert

  if (!gl.getProgramParameter(shaderProgram, gl.LINK_STATUS)) {
    alert("Unable to initialize the shader program: " + gl.getProgramInfoLog(shader));
  }

  gl.useProgram(shaderProgram);

  vertexPositionAttribute = gl.getAttribLocation(shaderProgram, "aVertexPosition");
  gl.enableVertexAttribArray(vertexPositionAttribute);
}
```

https://developer.mozilla.org/en-US/docs/Web/API/WebGL_API]

OpenGL rendering pipeline

vertex buffer

triangles

Vertex shader   Shape assembly   Geometry shader

{ vertices }

Tests and Blending   Fragment shader   Rasterization

frame buffer

https://open.gl/drawing

Summary

- WebGL provides Javascript API to easily render interactive 3D graphics

- Pros
  - ✦ No plug-in
    - ✢ Compare to Unity3D, Silverlight
  - ✦ Flexible and Scalable
  - ✦ Easy to integrate
    - ✢ Access to full DOM element

- Cons
  - ✦ Largely dependent on graphics unit

Thanks !