

# Introducción a WebGL

---

Katia Leal Algara

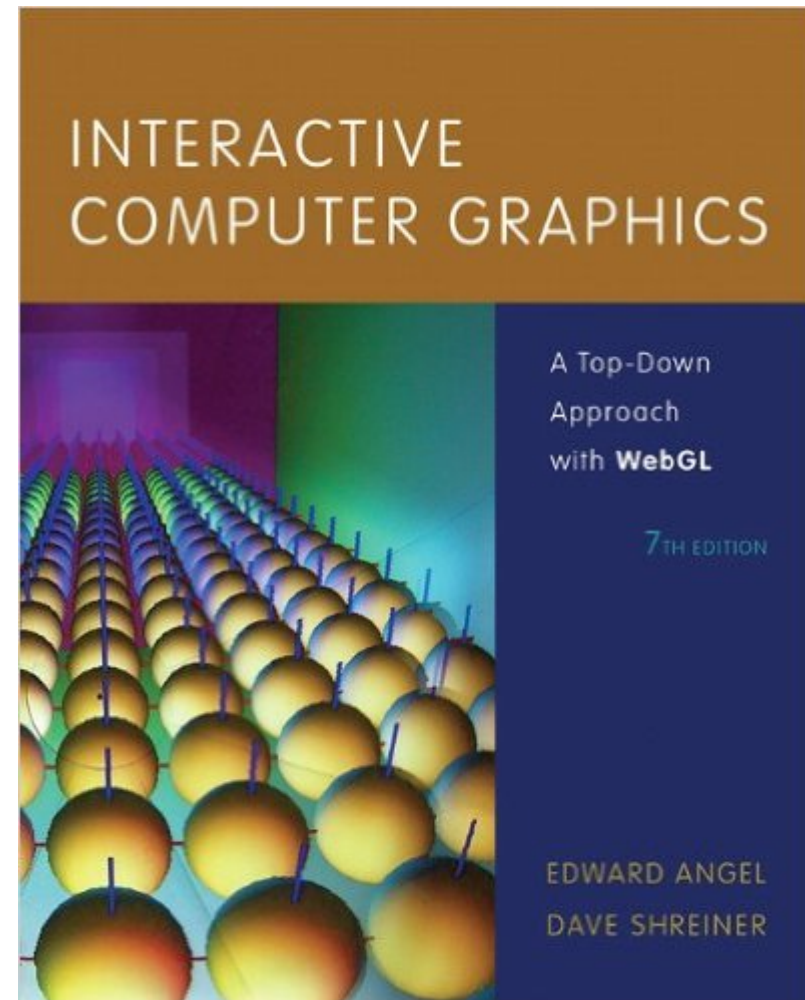
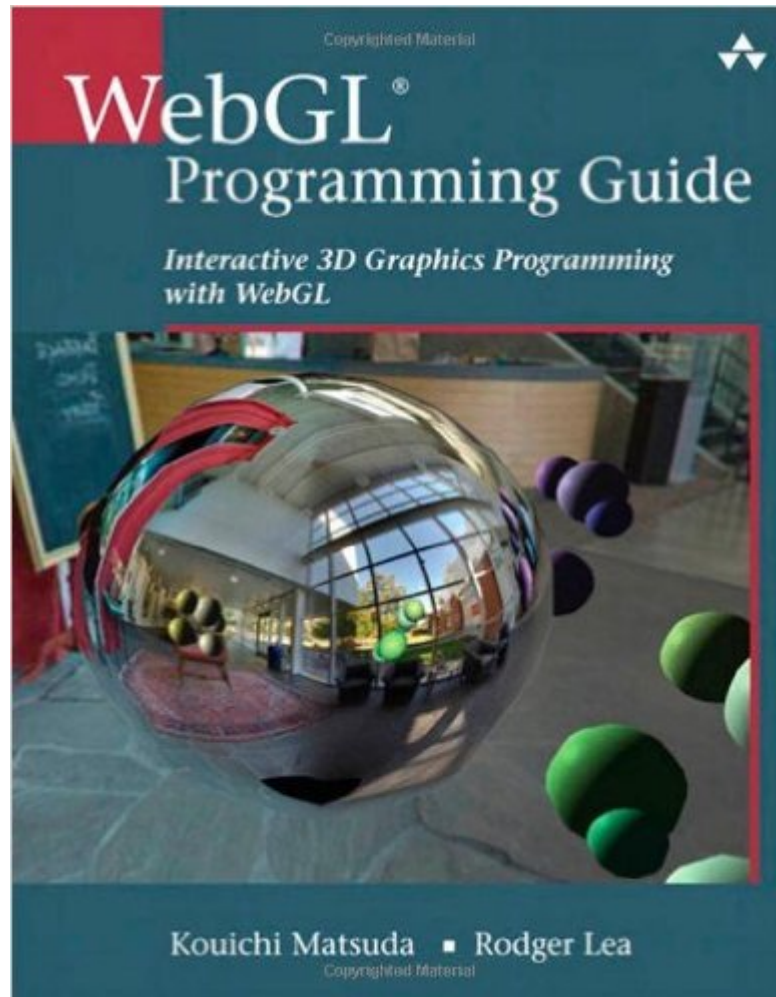
Web: <http://gsyc.urjc.es/~katia/>

Email: [katia.leal@urjc.es](mailto:katia.leal@urjc.es)

Dept. Teoría de la Señal y Comunicaciones y Sistemas Telemáticos y Computación (GSyC)  
Escuela Superior De Ingeniería De Telecomunicación (ETSIT)  
Universidad Rey Juan Carlos (URJC)



# Bibliografía WebGL



# Orígenes de WebGL: OpenGL

- **OpenGL (Open Graphics Library)** es una especificación estándar que define una API multilenguaje y multiplataforma para escribir aplicaciones que produzcan gráficos 2D y 3D.
- La interfaz consiste en más de 250 funciones diferentes que pueden usarse para dibujar escenas tridimensionales complejas a partir de primitivas geométricas simples, tales como puntos, líneas y triángulos.
- Fue desarrollada originalmente por Silicon Graphics Inc. (**SGI**) en 1992.
- Última versión estable 4.6, julio 2017.

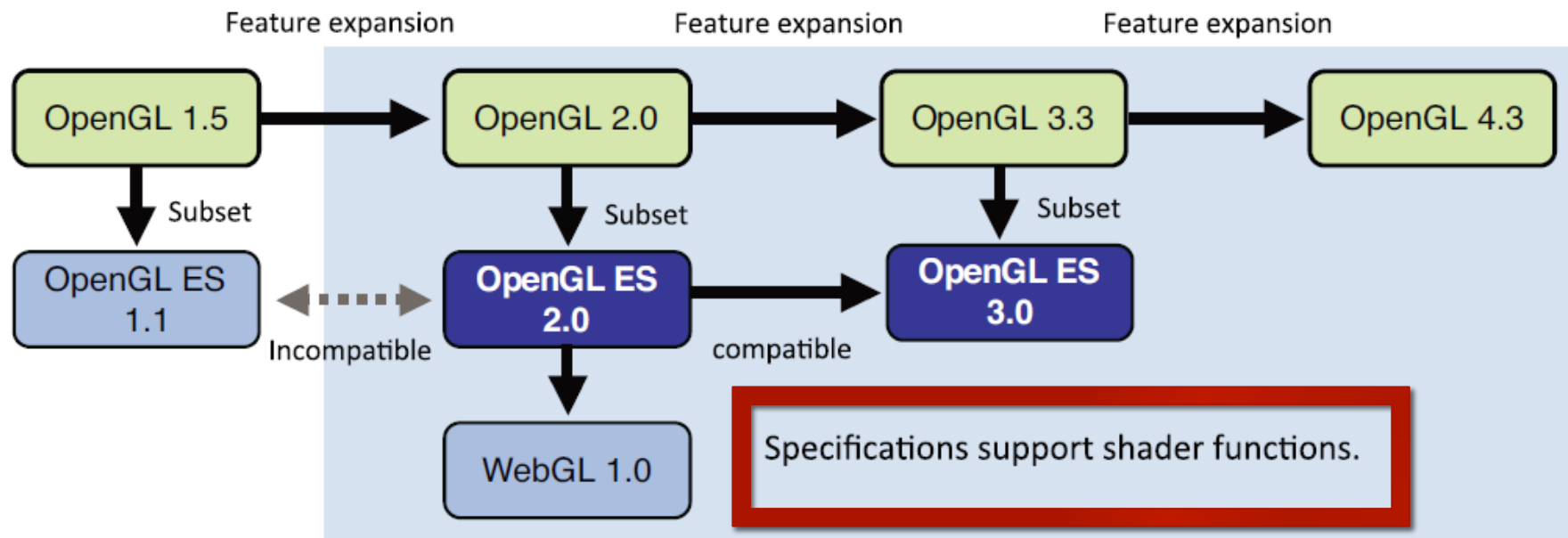


# Orígenes de WebGL: OpenGL ES 2.0

- **WebGL (Web Graphics Library)** es una especificación estándar que define una API implementada en JavaScript para la renderización de gráficos en 3D dentro de cualquier navegador web.
- Deriva de OpenGL ES (**E**mbdedded **S**ystems) originalmente desarrollada en 2004 y actualizada a la versión 2.0 en 2007, que es en la que se basa WebGL.
  - Se trata de una especificación que elimina muchas características innecesarias, lo que resulta en una *especificación ligera* con poder suficiente para generar gráficos potentes en 3D.
- Última versión estable 2.0, enero de 2017.
- WebGL está diseñado y gestionado por el consorcio de tecnología sin ánimo de lucro Khronos Group.



# Orígenes de WebGL: OpenGL ES 2.0



# Introducción a WebGL: *shaders*

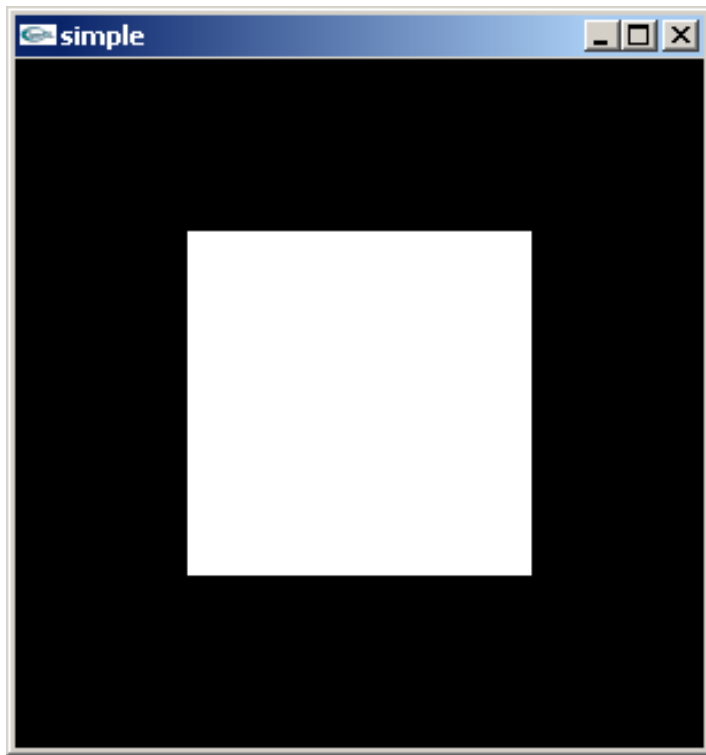
- OpenGL 2.0: capacidad para crear funciones *shader* o simplemente ***shaders***.
- Un *shader* es un código que permite programar efectos visuales sofisticados por medio de un lenguaje de programación especial parecido a C denominado ***shading language***.
  - **OpenGL shading language (GLSL).**
  - **OpenGL ES shading language (GLSL ES).**

# Introducción a WebGL: desarrollo

- Combinado con **HTML5** y **JavaScript**, hace los gráficos 3D accesibles a desarrolladores web.
  - Mejores interfaces de usuario.
- Desarrollo muchos más sencillo:
  - Antes: C/C++ y OpenGL o Direct3D.
  - Ahora: Editor texto y Navegador (Chrome).
- **Tarjetas soportadas:** [www.khronos.org/webgl/wiki/BlacklistsAndWhitelists](http://www.khronos.org/webgl/wiki/BlacklistsAndWhitelists)
  - Intel Corporation Device 5912.
  - NVIDIA Corporation GK106 [GeForce GTX 645 OEM].
- **Ejemplos:** <http://webglsamples.org/>.

# Un ejemplo sencillo

**OpenGL**



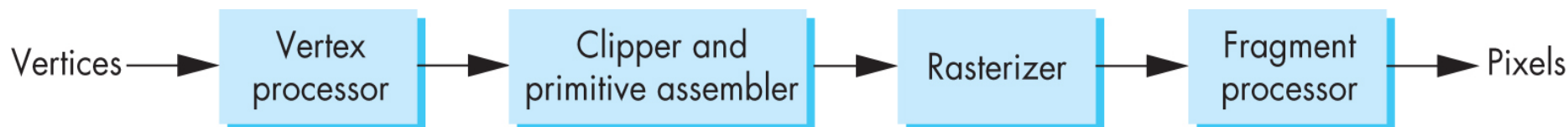
**WebGL**





# Introducción a OpenGL

- El rendimiento se obtiene utilizando la GPU en lugar de usar la CPU.
- Podemos programar la GPU con unos programas especiales llamados, *shaders*.
- Nuestra aplicación debe suministrar los datos para que trabaje la GPU.
- La GPU realiza todo el trabajo pesado:



## Funcionamiento de una aplicación WebGL/OpenGL

- WebGL/OpenGL trabaja en un bucle infinito:
  - Sitúa elementos en la escena(puntos, líneas, polígonos,..).
  - Describe la cámara (posición, orientación, *field of view*).
  - Atiende los eventos del teclado (*keyboard events*).
  - Dibuja la escena.

## Estado de una aplicación WebGL/OpenGL

- WebGL/OpenGL tiene un estado:
  - El programa OpenGL tiene multitud de posibles configuraciones.
  - La configuración actual se almacena en el estado de OpenGL.
  - Los comandos de OpenGL afectan al estado del programa.

# Tipos de primitivas

- GL\_POINTS
- GL\_LINE
  - {S | \_STRIP | \_LOOP}
- GL\_TRIANGLE
  - {S | \_STRIP | \_FAN}
- GL\_QUAD
  - {S | \_STRIP}

# Tipos de primitivas

v0      v2      v4

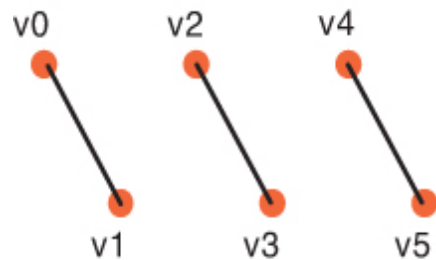


    v1      v3      v5



gl.POINTS

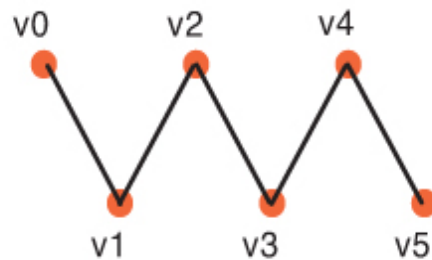
v0      v2      v4



v1      v3      v5

gl.LINES

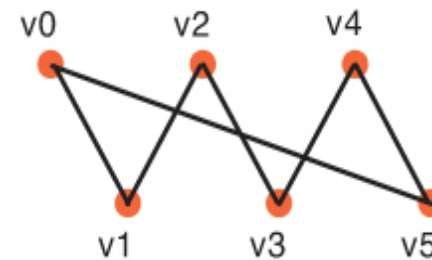
v0      v2      v4



v1      v3      v5

gl.LINE\_STRIP

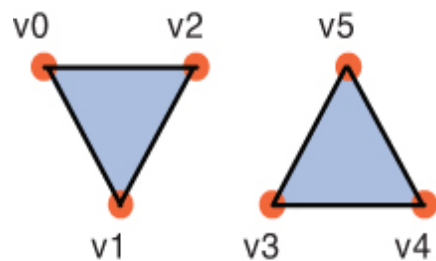
v0      v2      v4



v1      v3      v5

gl.LINE\_LOOP

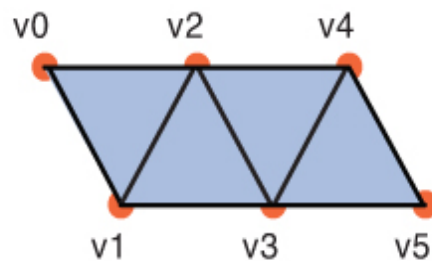
v0      v2



v1      v3      v4      v5

gl.TRIANGLES

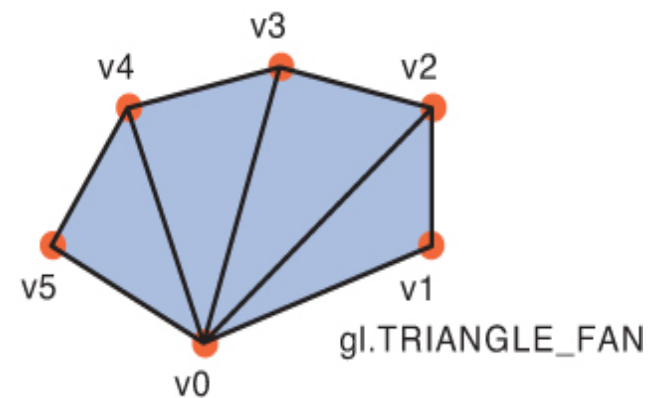
v0      v2      v4



v1      v3      v5

gl.TRIANGLE\_STRIP

v3      v2



v4      v1      v5      v0

gl.TRIANGLE\_FAN

# Estructura de una aplicación WebGL

- Las páginas web dinámicas se crean utilizando 3 lenguajes:
  1. HTML5 (Hypertext Markup Language).
  2. JavaScript.
  3. GLSL ES: este último normalmente se escribe en JavaScript, por lo que la estructura de ficheros de las páginas WebGL es básicamente la misma que la de una página web estándar.

# Estructura de una aplicación WebGL

