

# Desarrollo de un Sistema Gráfico

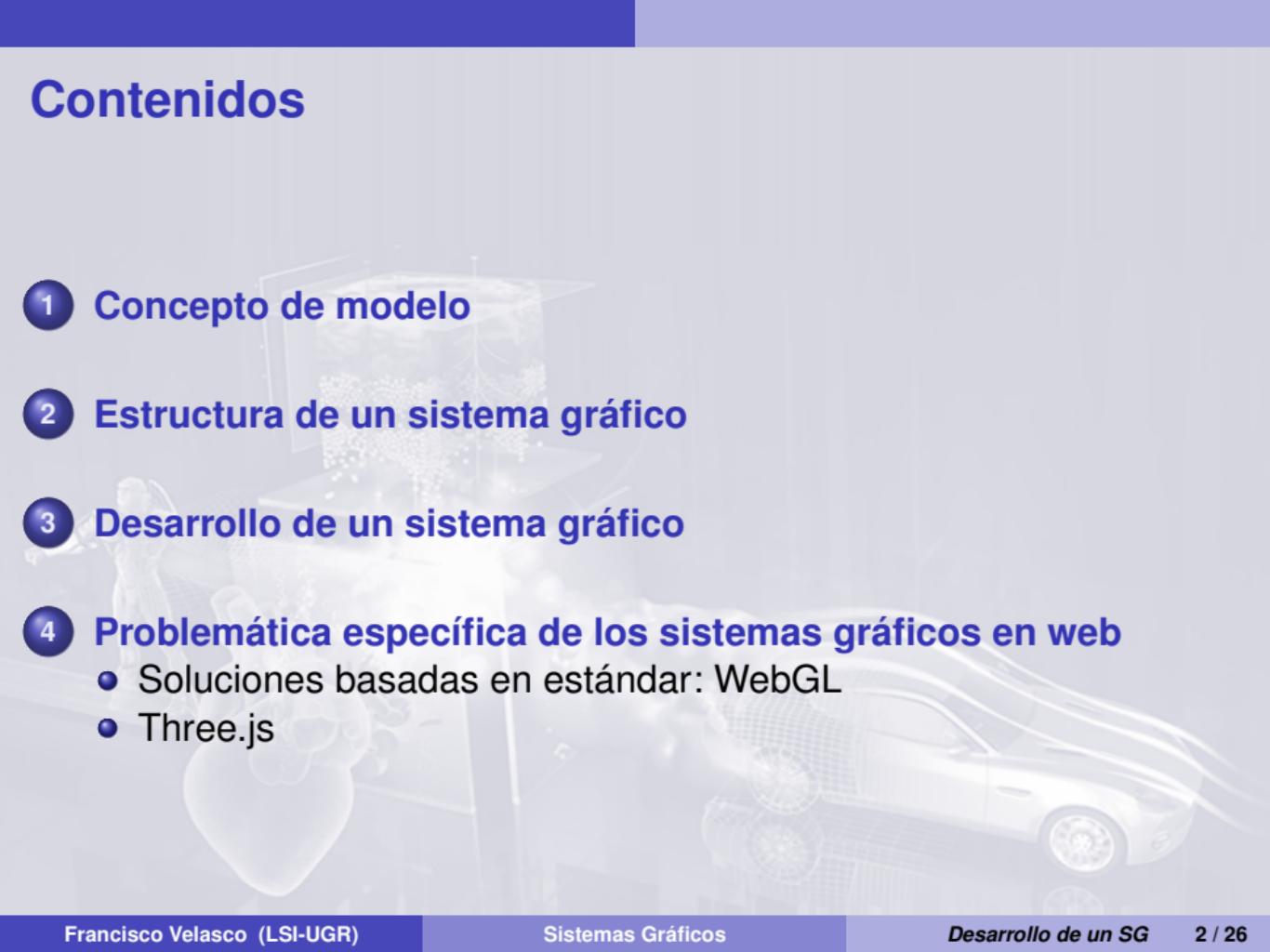
Francisco Velasco Anguita

Dpto. Lenguajes y Sistemas Informáticos  
Universidad de Granada

Sistemas Gráficos

Grado en Ingeniería Informática  
Curso 2017-2018

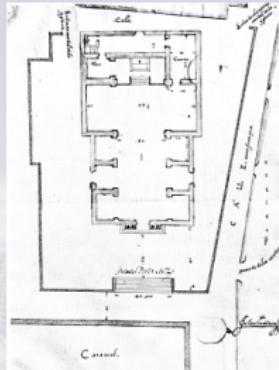
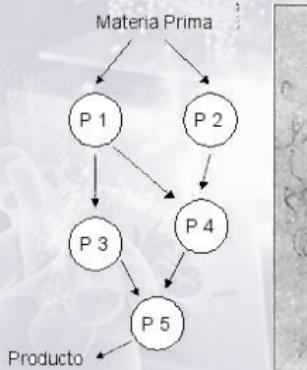
# Contenidos

- 
- 1 Concepto de modelo
  - 2 Estructura de un sistema gráfico
  - 3 Desarrollo de un sistema gráfico
  - 4 Problemática específica de los sistemas gráficos en web
    - Soluciones basadas en estándar: WebGL
    - Three.js

# Concepto de modelo

- ¿Qué es un modelo?

- ▶ Representación de características relevantes de una entidad
- ▶ La entidad puede ser real o imaginaria
- ▶ Permite *trabajar* con el modelo en vez de hacerlo con la entidad

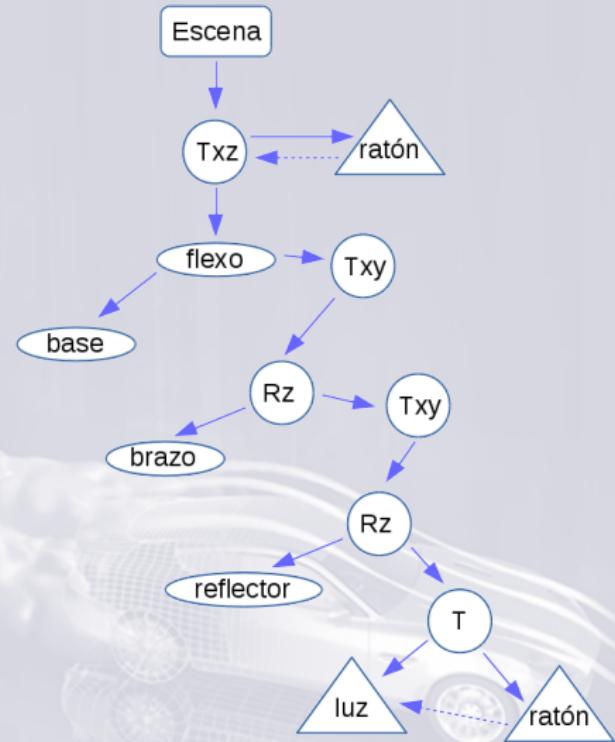


# Concepto de modelo



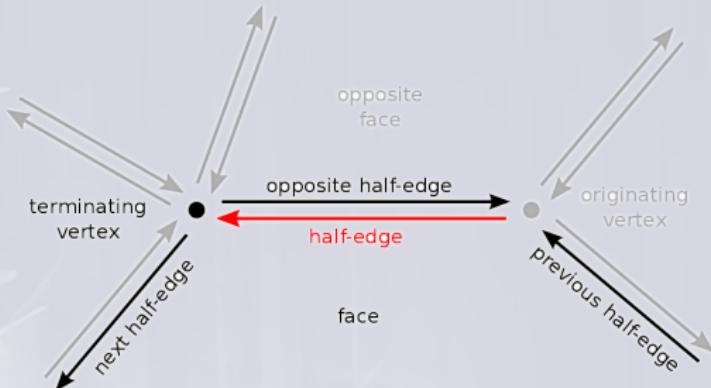
# Ejemplos de modelos

## Grafo de Escena



# Ejemplos de modelos

## Malla de Polígonos

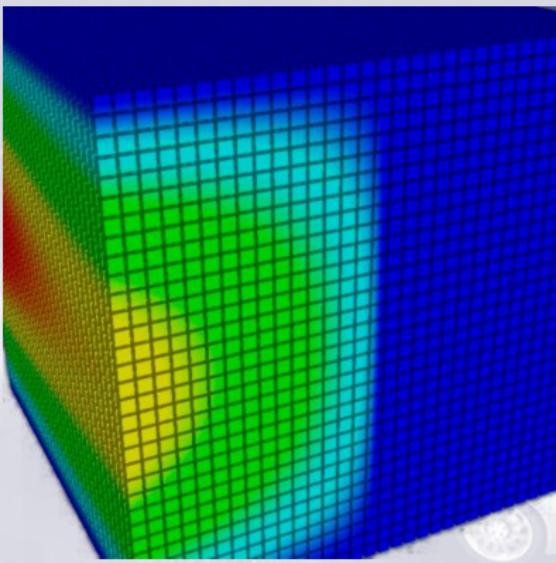


Representado por:

- Lista de semi-aristas aladas
- Lista de vértices
- Lista de caras

# Ejemplos de modelos

## Volúmenes



# Ejemplos de modelos

## Escenarios

- Se representan con una estructura de datos

Por ejemplo: Pieza tablero[8][8] ;



- A partir de esa representación se genera la escena

Por ejemplo: escena.add (tablero[i][j].getGeometria());

- El motor de rendering se encarga de generar la imagen



# Estructura de un sistema gráfico

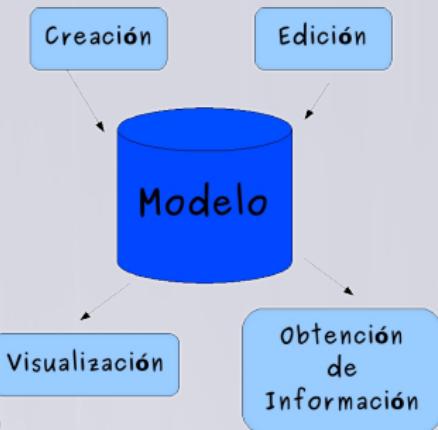
- **Creación**

- ▶ Procedural
- ▶ Carga de archivos
  - ★ Digitalización
  - ★ Modelado
- ▶ Interactivo
  - ★ Procesamiento de eventos: ratón, teclado

- Visualización

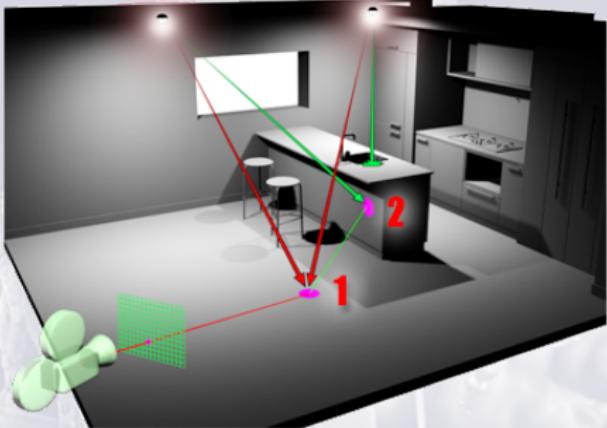
- Edición

- Obtención de información

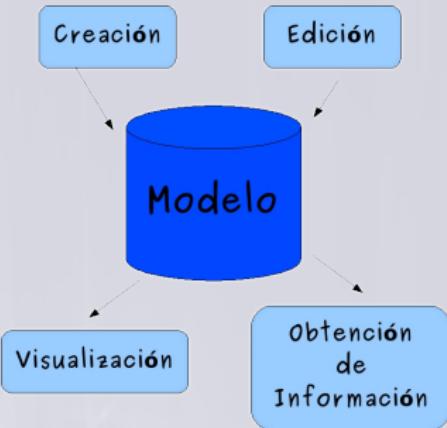


# Estructura de un sistema gráfico

- Creación
- Visualización



- Edición
- Obtención de información



# Estructura de un sistema gráfico

- Creación

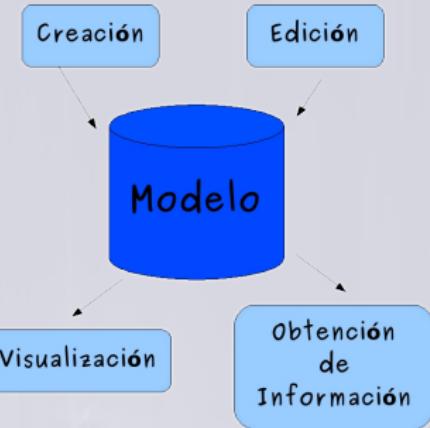
- Visualización

- Edición

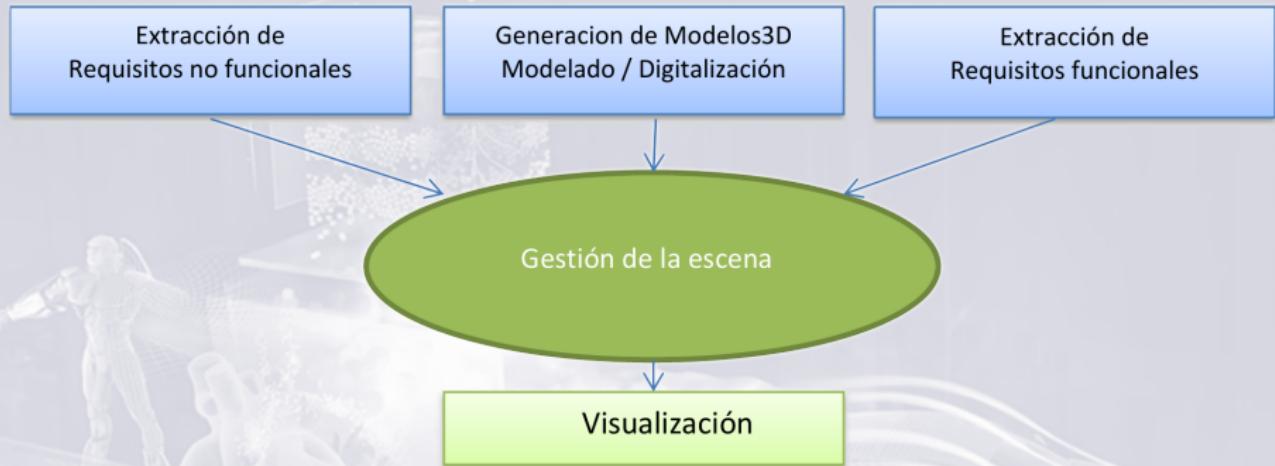
- ▶ Objeto - Tiempo → Animación
- ▶ Objeto - Objeto → Colisiones
- ▶ Objeto - Usuario → Pick
  - ★ Procesamiento de eventos: ratón, teclado

- Obtención de información

- ▶ Generación de listados



# Desarrollo de un sistema gráfico



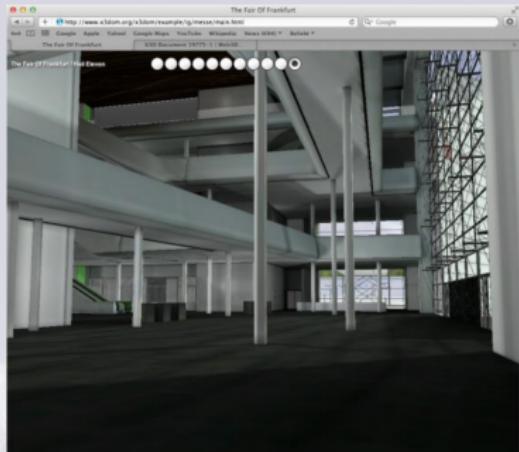
# Ejercicio



- ¿A qué bloque de la figura anterior pertenece cada situación?
  - 1 El cliente nos pide un juego sobre regatas náuticas.
  - 2 Se modelan en 3D Studio los barcos.
  - 3 Se digitalizan algunos barcos para tener modelos *exactos*.
  - 4 Los modelos de mar y viento son físicamente correctos.
  - 5 Se quieren hacer versiones para PC y dispositivos móviles.
  - 6 Se crea una regata donde intervienen unas características de mar y viento concretas y unos barcos ya preseleccionados.
  - 7 Si dos barcos colisionan muestran sus desperfectos.
  - 8 Se realizan varios renders con distintos efectos visuales.
  - 9 Los barcos están a distinto detalle según su cercanía a la cámara.
  - 10 Se nos pide una versión para la tarjeta GeForce GTX 1070.

# Problemática específica de los sistemas gráficos en web

- Desarrollar sistemas gráficos para la web es una tarea compleja
- Supone ejecutar algoritmos complejos en plataformas con recursos hardware muy dispares
- No nos referimos a:
  - ▶ Renderización remota de gráficos
  - ▶ Descarga de modelos de un servidor
- Se trata de la visualización 3D en el marco de un navegador web



Escena 3D sobre Safari

©Fraunhofer IGD, Darmstadt

# Visualización 3D en web

## Problemática específica (1)

- Volumen de información y velocidad de transferencia
  - ▶ Arquitectura cliente / servidor
  - ▶ Separados cientos o miles de kilómetros
  - ▶ Con una velocidad muy variable. A veces, muy limitada
  - ▶ Un modelo 3D puede tener miles o millones de polígonos
- Altos requerimientos computacionales
  - ▶ La visualización 3D requiere bastantes recursos hardware
  - ▶ No todo usuario de un navegador dispone de ellos

# Visualización 3D en web

## Problemática específica (y 2)

- Entorno de ejecución restringido al navegador web
  - ▶ Memoria limitada
  - ▶ Sin acceso directo al sistema de ficheros,
  - ▶ Ni a las bibliotecas existentes en el sistema
- Diversidad de plataformas
  - ▶ Sistemas operativos
  - ▶ Navegadores



# Tecnologías

- En los últimos años
  - ▶ VRML
    - ★ Estándar no aceptado nativamente por todos los navegadores
    - ★ Requiere de plugins, normalmente de pago, para su funcionamiento
  - ▶ Applets Java
    - ★ Se ejecuta sobre la máquina virtual Java, se evitan las restricciones que impone el navegador
    - ★ Pero está limitado a 64 MB y no soporta más de 250.000 polígonos
- Últimamente se han desarrollado diversos estándares Web
  - ▶ Motivado por la pérdida del monopolio de Internet Explorer y Safari
  - ▶ Facilitado por:
    - ★ La aparición de HTML5, incluye nativamente un elemento canvas
    - ★ La definición de WebGL, biblioteca JavaScript que accede a las capacidades OpenGL del cliente

# WebGL



- <https://www.khronos.org/webgl/>
- Aparece con el apoyo de Google, Apple y Mozilla
  - ▶ Microsoft se sumó más tarde
- Es una API
  - ▶ Se accede mediante interfaces en JavaScript, no mediante etiquetas HTML
- Está basado en OpenGL ES 2.0 (OpenGL Embedded System)
- Utiliza GLSL
  - ▶ El uso de shaders es obligatorio
- Puede convivir con otro contenido HTML
- Es multiplataforma
- Es gratis

# WebGL: Desventajas

- Requiere programar a bajo nivel
- No se dispone de primitivas geométricas
  - ▶ Se requiere usar Buffer Objects
- No se dispone de matrices de transformación
  - ▶ Hay que implementarlas
- No se disponen de materiales ni de fuentes de iluminación
  - ▶ Hay que diseñarlos e implementarlos
- Hay que programar Shaders para visualizar
  - ▶ Obligatoriamente

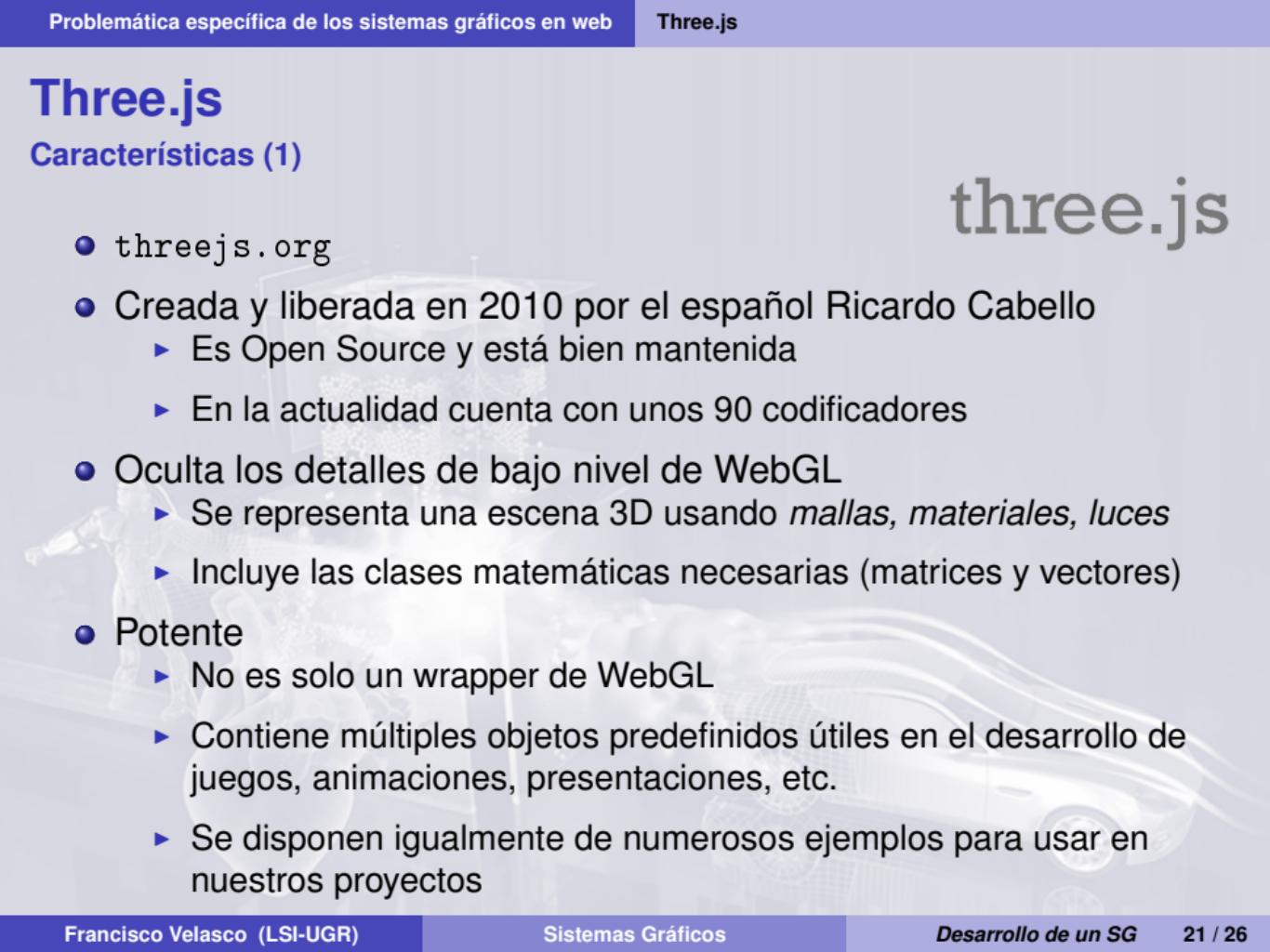
(Ver el ejemplo subido a Prado)

# WebGL: Ventajas

- Proporciona flexibilidad al desarrollar sistemas gráficos
- Y eficiencia gracias al uso de buffer objects y shaders
- **Sin embargo ...**
  - ▶ Las desventajas comentadas tienen mucho peso
- **Alternativa:** Usa una capa sobre WebGL
  - ▶ Escribir tu propia biblioteca de alto nivel, definiendo clases para representar geometría, materiales, etc.
  - ▶ Usar una biblioteca de las ya existentes
- La biblioteca líder en este campo es **Three.js** ([threejs.org](http://threejs.org))

# Three.js

## Características (1)

A faint background image of a 3D rendered car interior, featuring a steering wheel and dashboard, serves as the backdrop for the slide.

# three.js

- [threejs.org](http://threejs.org)
- Creada y liberada en 2010 por el español Ricardo Cabello
  - ▶ Es Open Source y está bien mantenida
  - ▶ En la actualidad cuenta con unos 90 codificadores
- Oculta los detalles de bajo nivel de WebGL
  - ▶ Se representa una escena 3D usando *mallas, materiales, luces*
  - ▶ Incluye las clases matemáticas necesarias (matrices y vectores)
- Potente
  - ▶ No es solo un wrapper de WebGL
  - ▶ Contiene múltiples objetos predefinidos útiles en el desarrollo de juegos, animaciones, presentaciones, etc.
  - ▶ Se disponen igualmente de numerosos ejemplos para usar en nuestros proyectos

# Three.js

## Características (2)

- Rápida
  - ▶ Usando buenas prácticas de programación, con un alto rendimiento
- Robusta
  - ▶ Dispone de chequeo de errores, excepciones y warnings en consola que facilita el desarrollo
- Soporta interacción (picking)
- Soporta formatos de archivo 3D
  - ▶ Formatos ASCII de los programas de modelado 3D
  - ▶ Un formato específico de Three.js, JSON
- Orientada a objetos, extensible
- Fácil de usar, y de aprender

(Ver el ejemplo subido a Prado, y compararlo con el de WebGL)

# Three.js

## Lo que NO es

- No es un motor de juegos
  - ▶ No incluye billboards, avatares, física, etc.
- No incluye soporte para redes
  - ▶ Necesario para hacer juegos multijugador
- No es un framework de aplicaciones
  - ▶ No dispone de estructuras para plantillas, gestión de sesiones, etc.
- No es un entorno integrado
  - ▶ No dispone de herramientas para el desarrollo completo de aplicaciones

# Ejemplos realizados con Three.js

- Video clip



<http://www.ro.me>

- Juegos



<http://cwar.de/pinball/simpleball.html>

- Configurador de coches



<http://carvisualizer.plus360degrees.com/threejs>



<http://hexgl.bkcore.com>

# Three.js

## Recursos

- **Página oficial:** <http://threejs.org/>

Incluye la biblioteca para descargársela, la documentación y bastantes ejemplos

- **Bibliografía**

- ▶ J. Dirksen;

**Learning Three.js: The JavaScript Library for WebGL;**  
recurso electrónico en [biblioteca.ugr.es](http://biblioteca.ugr.es)

Ejemplos en:

<https://github.com/josdirksen/learning-threejs>

- ▶ J. Dirksen;

**Three.js Essential;**

recurso electrónico en [biblioteca.ugr.es](http://biblioteca.ugr.es)

Ejemplos en:

<https://github.com/josdirksen/essential-threejs>

# Desarrollo de un Sistema Gráfico

Francisco Velasco Anguita

Dpto. Lenguajes y Sistemas Informáticos  
Universidad de Granada

Sistemas Gráficos

Grado en Ingeniería Informática  
Curso 2017-2018

Parte de este material ha sido realizado en colaboración con Francisco Javier Melero Rus