# Especificación del trabajo

En este capítulo se presenta la especificación del trabajo, con una estructura y contenidos **inspirados** en los criterios y recomendaciones que establece la norma UNE 157801:2007 - “*Criterios Generales para la elaboración de proyectos de Sistemas de Información*”.

A lo largo del documento se utilizarán términos y acrónimos que estarán en negrita y cuya descripción aparecen en el **apartado 8** (*Definiciones y abreviaturas).*

## Introducción

La asignatura de Programación de Aplicaciones Gráficas (PAG) del Grado de Ingeniería Informática, tiene una curva de aprendizaje muy pronunciada, debido que abarca todo el proceso para la generación de gráficos.

La creación de gráficos necesita de tres partes esenciales para la correcta creación de la imagen que se quiere mostrar. Hace falta una geometría correcta, qué esté correctamente cargada en la Unidad de procesamiento gráfico (**GPU**); un shader OpenGL Shading Language (**shader** **GLSL**) correcto, bien compilado y cargado en la GPU; y establecer correctamente los parámetros de Open Graphics Library (**OpenGL**). La dificultad reside en que no se puede probar cada pieza por separado antes de visualizar el resultado final ya que las tres piezas son imprescindibles para poder ejecutar la aplicación gráfica.

En la asignatura de PAG se lleva a cabo el desarrollo de una aplicación gráfica que engloba las tres partes, pero no se pueden probar hasta la finalización de las tres, por lo tanto, si hay algún error en alguna de las tres partes, no se puede apreciar hasta la finalización de la aplicación.

En este Trabajo de Fin de Grado se pretende desarrollar una herramienta multiplataforma que permita al usuario probar directamente los Shaders GLSL que esté desarrollando, sin necesidad de haber escrito las otras dos piezas de software necesarias. La herramienta proporciona un entorno donde poder escribir o cargar los Shaders que el usuario quiera probar y aplicarlos sobre geometrías predefinidas y con configuraciones de parámetros predefinidas, así como comprobar la validez sintáctica de los Shaders escritos.

## Antecedentes y estado del arte

Existen una serie de aplicaciones que resuelven parcialmente el problema, entre ellas:

* **Shdr**: [1] es un editor online ESSL (GLSL), funciona como editor, visualizador y validador de shaders todo con **WebGL** creado porThibaut Despoulain.

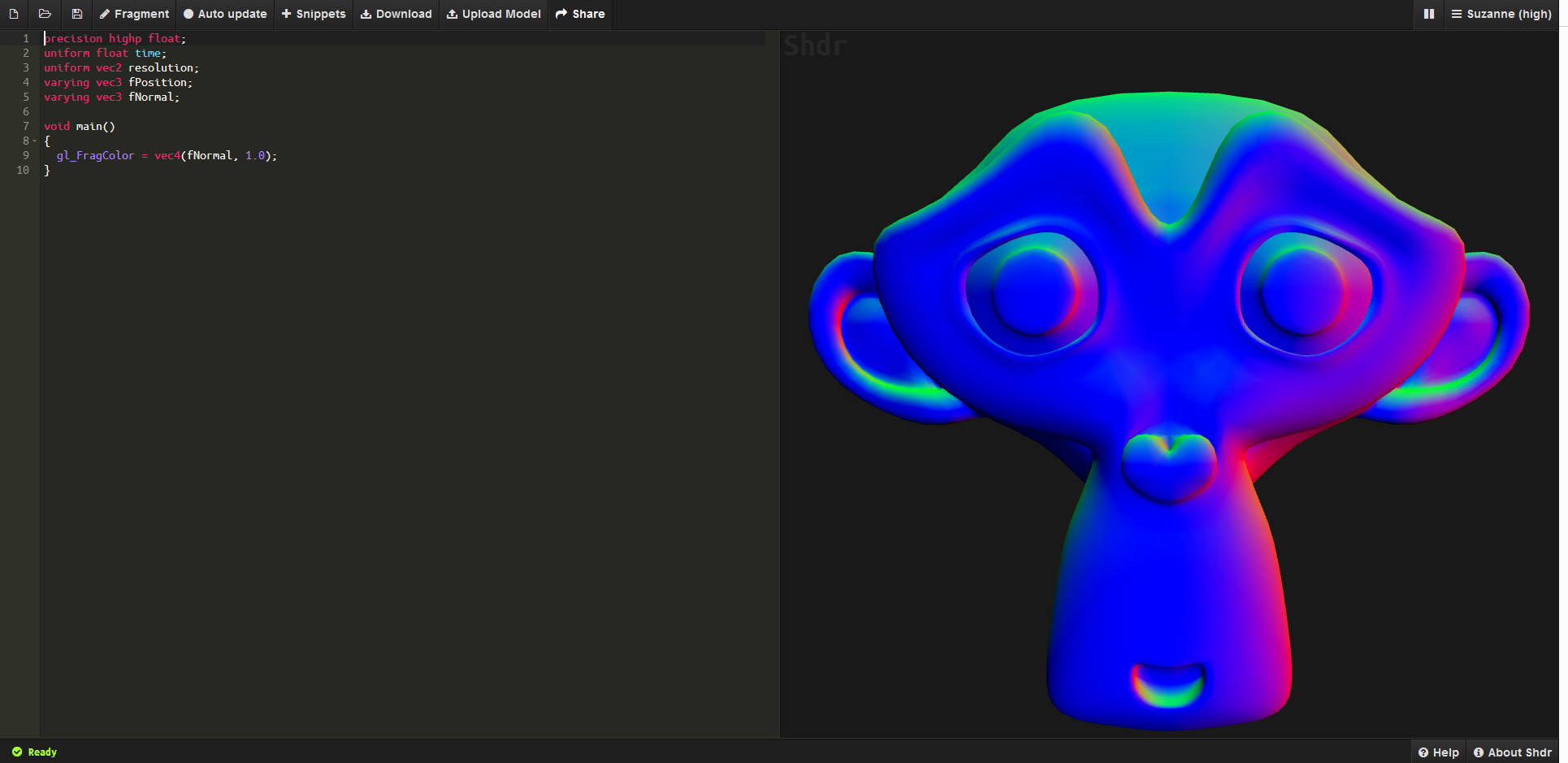


Ilustración 1 Aplicación Shdr

* **Kick.js**: [2] Es un motor gráfico montado en WebGL para navegadores modernos, que además posee un editor de shaders.

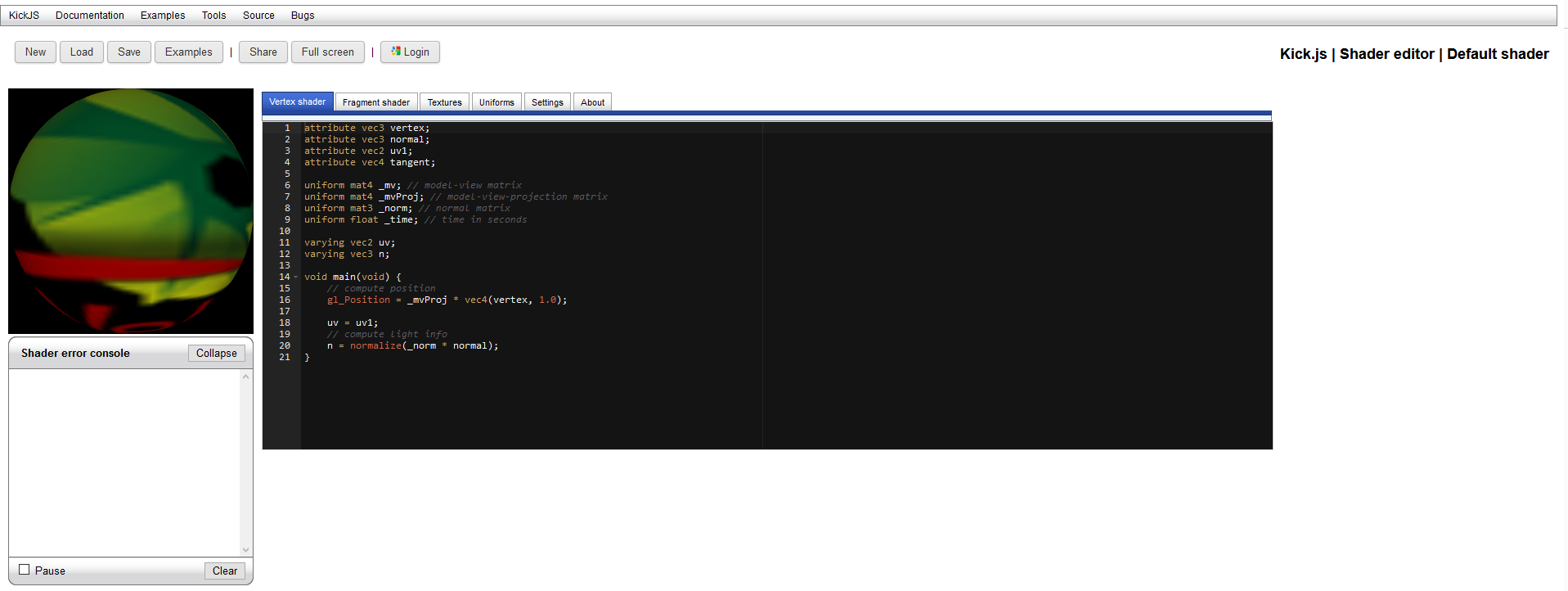


Ilustración 2 Aplicación Kick.js

Estas aplicaciones online presentan una serie de inconvenientes importantes:

* No están totalmente adaptadas a la asignatura PAG del Grado de Ingeniería Informática y no permiten usar características avanzadas de OpenGL, (no disponibles en WebGL).
* Son complicadas y poco intuitivas.

En este Trabajo de Fin de Grado se propone una herramienta basada en OpenGL 4.5, para la edición, visualización y validación de los shaders GLSL.

# Objetivos del trabajo

* Desarrollar un prototipo que permita cargar y compilar shaders GLSL, compilarlos y mostrar si son correctos desde el punto de vista sintáctico.
* Implementar algunas geometrías predefinidas, sobre las que se puedan aplicar los shaders cargados, para ver si el resultado obtenido es el que se buscaba.
* Visualizar el resultado de la ejecución de un shader program.
* Mostrar resultados intermedios en forma gráfica, para facilitar la prueba de los shaders, como por ejemplo los cube maps en el caso de shaders para simular reflexión o los shadow maps en el caso de shaders para simular sombras arrojadas.

# Bibliografía

|  |  |
| --- | --- |
| [1] | T. Despoulain. [En línea]. Available: http://bkcore.com/blog/3d/shdr-online-glsl-shader-editor-viewer-validator.html. |
| [2] | kickjs. [En línea]. Available: http://www.kickjs.org/example/shader\_editor/shader\_editor.htm. |