MG Engine

Miguel Gutiérrez

Explicación de la arquitectura del motor

El motor se construye en C++ a partir de las bibliotecas de código: *SDL*, *OpenGL Toolkit* y *GLM*, además de la biblioteca estándar. Basa su estructura en el patrón de diseño **ECS (Entity, Component, System)**:

- **Kernel**: Contiene el bucle principal del motor, ejecuta en cada *frame* las tareas que componen el funcionamiento de las aplicaciones.
- **Entity**: Son los elementos del juego. Contienen componentes que les dan funcionalidad.
- **Component**: Por defecto hay 4 tipos de componentes:
    - R*ender* para que el objeto tenga una malla a renderizar.
    - *Event Listener* para que el objeto escuche a eventos del motor.
    - *Controller* para añadir comportamiento a las entidades.
    - *Transform* matriz de transformación del objeto.
- **System**: Contienen las tareas que se tienen que ejecutar en el kernel. Estas tareas tienen una lista de componentes a los que aplicar distinta lógica en cada vuelta del bucle. Por ejemplo, la tarea de *render* renderizará las mallas de cada *render component* existente.

El kernel de tareas, las entidades y los sistemas están contenidos en una clase **Scene** que actúa de nexo entre todas ellas. Añade al kernel las tareas de los sistemas que deben ser ejecutadas, contiene un mapa de las entidades creadas y una ventana de la que recibir los inputs y donde renderizar las entidades.

Observaciones

Debido a la necesidad de sobreescribir los componentes de *Event Listener* y *Controller* para cada proyecto que use este motor, se ha prescindido puntualmente de la idea de sistema y se han añadido unos métodos *add_controller* y *add_event_listener* a las entidades. De esta manera se evita un paso más y el usuario del motor puede añadir controladores y listeners sin tener que pasar por los sistemas.

Se requiere para cada proyecto que use este motor la inclusión de las bibliotecas: *GLM* para manipular los componentes *transform* de las entidades y *TinyXML 2* para poder leer los archivos .xml que contienen los datos de una escena.

Código y futuras mejoras en MGEngine.