SLB:

Un dibujo de un pizarrón blanco

Descripción generada automáticamente con confianza baja

C++ ENGINE –> SLB .

El SLB permite registrar funciones (clases, atributos y miembros). Entonces LUA tiene objetos. El SLB lo hizo Pepelu, un español.

Tendriamos que hacer una capa Wrapper para convertir las llamadas de C++ a C. Es mapear de C++ a C. SLB ya lo hace y nos lo ahorra.

SLB crea un container para la clase. Tiene un objeto SLB::Class pero no es una clase C++, es un objeto. Esta class es un container (tiene una clase).

Texto

Descripción generada automáticamente

Tiene atributos, un constructor, vainilla.

Texto

Descripción generada automáticamente

Declaración de un objeto que se llama LogicManager, que es una SLB::Class container de LogicManager.

Nosotros tenemos una clase C++, y vamos a meter la clase en un cajon contenedor.

Una caricatura de un pizarrón blanco

Descripción generada automáticamente con confianza media

Al haberlo metido dentro, la clase hereda cosas de SLB (ya tiene todos los métodos para que se pueda publicar todo).

Codigo SLB

Tabla

Descripción generada automáticamente

Y una carpeta lua, donde esta todo lo del LUA. Nosotros ahora no nos preocupamos del LUA, SOLO de SLB.

## Vemos como va el codigo

SLB:: es el namepsace

* SLB::Manager 🡪 Es la maquina virtual. Esto iniciliza el SLB y la amquina virtual que lleva dentro
* SLB::Script (le pasamos por referencia el manager).
  + El LUA funcionaba como una pila. Yo le paso el manager pq ahí es donde están todas las declaraciones de variables. UN SOLO MANAGER. Y UN solo script. Y así tenemos todo ahí.
  + S.doFile() pero si queremos otro no tendría visibilidad común. **Entonces SOLO 1!!**!
  + El SLB es más moderno. El doFile ya tiene excepciones.

Texto

Descripción generada automáticamente

Si queremos gestor de expciones así

Se pueden publicar de una clase todo (pero se publica MANUALMENTE). Puedo tener una clase que tenga 20 miembros de los que quiero publicar 4. Ejemplo:

* AudioManager: pero yo tengo un montón de miembros pero yo para los artistas, le publico el audio manager y el PlaySound y todo los demás NO.
* Para publicar

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Publica la variable. La clase class tiene un miembro qque se llama

* + Comment
  + Constructor
  + Set

Texto

Descripción generada automáticamente

Esto se llama **Method Chaining** (en java se usa mucho).

Si cada llamada devuelve la clase entonces voy llamando a una funciion y esa función me devuelve la clase en si pues sobre el resultado vuelvo a llamar a la clase.

OJO: Hay que llevar cuidado con el orden de las llamadas en este método Chaining. NO debe haber dependencias entre sí

.Comment 🡪 Desde el LUA pueda hacer un

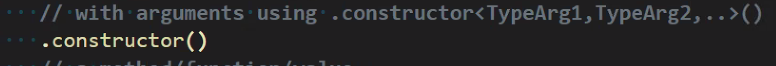
.Constructor 🡪 Publica el constructor

Está publicando esto:

Texto

Descripción generada automáticamente

Si tuviera argumentos seria así

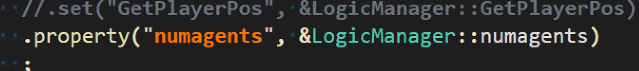


Los miembros se publican con .set

Texto

Descripción generada automáticamente

Publicar atributos (deben ser publicos):



## PASAMOS AL LUA:

Hay que poner siempre el include

Texto

Descripción generada automáticamente

Un **print** hace esto



Texto

Descripción generada automáticamente

Internamente sabe esto. El SLB nos permite hacer un print de la clase.

Llamamos a un constructor

Imagen que contiene golpear, pelota, jugador, sostener

Descripción generada automáticamente

La sintaxis es con 2 puntos pa llamar a una función

Lm:getpayerlife

 (vale basura al inicio pq nadie iniciliaza)

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente con confianza media

Si llamo al RespawnPlayer ya se iniciliza la vida a 100.

Acceso a atributos

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Llamar a funciones



Yo voy a llamar a nombres de eventos sin saber si hay alguien escuchándolo

Texto

Descripción generada automáticamente Interfaz de usuario gráfica, Texto, Chat o mensaje de texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Si alguien se equivoca de nombre. Este está mal:

Texto

Descripción generada automáticamente

Se captura con la excepción

Texto

Descripción generada automáticamente

A veces no queremos hacer una llamada si no hay nadie suscrito al evento. Hay una llamada que se llama exists.

**NUNCA** debemos hacer un s.doString() pq no sabemos si eso está bien

**DEBEMOS**: manager tiene las clases, Scripts tiene las funciones

El script (la s) le puedo hacer una llamada a exists para ver si eso existe.

Texto

Descripción generada automáticamente

Y te devuelve en p1 false pq no existe y en p2 true si existe

Un gestor de expciones NO es rápido, es mejor hacer llamadas a ciegas. Entonces antes metemos un exists y si está bien la primera vez ya sabemos que lo podemos llamar las siguientes veces.

## Ejercicio de prueba:

COGEMOS LA DEMO y declaramos un par de miembros ya tributos mas a nivel de clase.

1. Crear miembros + atributos nuevos
2. Publicarlos a SLB
3. Usarlos desde LUA

Si un programador empieza a publicar todo MAL. El scripting no se hace porque sí.

La idea es que el Engine es para tareas repetitivas y rápidas. Y esto es para control de sesión, de la partida, control de audio o cosas así. Es algo típico fliparse con esto.

## Inconveniente SLB

El SLB tiene un inconveniente (mañana lo integramos con Juan).

Los mensajes de Warning y Error son muy raros.

## Comentario

Lo hizo Pepelu. Ahora mismo el tio este empezó haciendo demos y tal Jose Luis Hidalgo Valiño

[www.pplux.com](http://www.pplux.com)

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamente

Sketfab es para dibujo vectorial. Este trabajaba ahí y sketfab lo absorbió epicGames.

Megascans: librería de texturas de alta resolución. La compro tamb epic games. Y estaba en Madrid.

# Segunda parte: Con estas piezas como es el motor de lógica de un videojuego

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Podemos ver en un motor los eventos que tiene, la sintaxis y tal.

Lenguaje de programación del BIWARE (extensión del LUA para videojuegos).

Object Factories:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

No se pueden hacer ENUMS, lo típico es hacer los enums con variables.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Crear nuestro propio Sistema de Eventos (doString con nombre de evento) si hay alguien escuchando GUAY si no pues GUAY tamb (onStart, OnPlayerKill, on…) Los nombres de los eventos podemos crearlos proceduralmente. Como el código ASCII es código fuente.

Ej: Hay una caja. OnENTER\_ rio.

OnStays prohibidos. (Ontick). Yo voy a ser envenenado. El programador tonto meteria un update { if (envenedado) tal } MAAAAAL

En LUA es muy lento, hay Estados Alterados. Lo q se publica a LUA son dos llamadas ActivateState y un RemoveState.

* OnEnterCharco 🡪 Activa estado alterado
* OnExitCharco 🡪 Desactiva

## Pizarra compartida

Publicamos los estados alterados a pizarras compartidas.

* Write
* Read

Texto

Descripción generada automáticamente

Lo hago con pizarras compartidas.

## En un motor de scripting se hace timers.

Publicamos a LUA el timer para que desde LUA pueda llamar a algo después de X tiempo.

Cualquier motor debería tener un SECUENCIADOR

Ejemplo: secuenciador para controlar la explosión de las granadas. De aquí a 4 segundos explota. O al recargar el rifle.

Un dibujo de un pizarrón blanco

Descripción generada automáticamente con confianza media

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Otro ejemplo:

Se puede hacer en LUA una banda de música procedural

Cosas típicas de LUA:

* Cambiar de cámara al entrar en volume

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

* Fabricar destruir enemigos
* Teleports
* Activar / desactivar IAs o nodos de BT
  + Ej: tenemos 3 dificultades de enemigos. Y tenemos BT con decoradores que miran la dificultad. En c++ tendrá un int level (Basic, médium, pro)
* Cambiar SFX

La clase del viernes es de 6 a 9 no??

2 clases para demostrar todo lo que hemos hecho en clase:

Individual : Tiene que usar BT y LUA (slb)

Juan ya lo sabe, nos va a ayudar a integrar el sistema de BT y SLB en el código. Eso no es el problema. Le dedicamos 20 min a decir: mi proyecto : un player con gestor de camra, controlador de triggers o BT chorra. Un diseño técnico para intentar

Miércoles y lunes para intentar implementarlo (EL ENTREGABLE). Currarnoslo. Una demo buena de C++ para sacarla y verla. Si hacemos eso las empresas ven que

NO HACER no se que plataformero (no tenemos colisiones).

BT para nuestro proyecto de luego!!

O algo de la diapo de Dani

Lo hacemos en clase, y lo entregamos en clase el LUNES!!!

No hay deberes de casa!

Alomejor hacemos enemigos de GOD con Blackboard o ejercicios de enemigos y soldados en desembarco de normandia. En total tenemos 6h.