Programación de Videojuegos 3D en Darkbasic (I)

La programación de videojuegos 3D por mucho tiempo ha sido una de las ramas de la programación en la cual se necesitaba para cumplir los proyectos una buena cantidad de personas interviniendo en estos además de que estas personas debian tener grandes conocimientos de programación Grafica y matematica.

Hace unos años atras cuando el programa español Div gamestudio estaba dando sus ultimos pasos, aparecio una nueva herramienta llamada Darkbasic, la cual permitia de una forma facil desarrollar juegos en 3d utilizando las direct X, pero sin que fuera necesario que el usuario supiera programar estas.

Hoy en dia existen 2 versiones de Darkbasic la profesional y la estandar, en este tutorial nos abocaremos a la versión Profesional, por contar esta con grandes mejoras respecto a la estandar (o clasica).

Fig1: Un videojuego de calidad profesional realizado en Darkbasic

¿Que queremos lograr con este tutorial?

El objetivo a la hora de realizar este tutorial es poder aprender los fundamentos básicos de la programación 3d. El objetivo a lo largo de los siguientes meses que se vaya realizando el tutorial es poder lograr que al finalizar el mismo, el lector tenga un panorama general de como crear juegos en 3d, se comenzará desde lo más básico, pero la idea es que cuando terminemos este tutorial, se realice un videojuego en 3D, aplicando lo aprendido a lo largo de los tutoriales. Es importante que el usuario tenga bien aprendida las instrucciones básicas del darkbasic, como por ejemplo los bucles (Do... Loop, While) las sentencias condicionales (if y Case) etc. En todo caso, se hará referencia a las instrucciones pero no nos detendremos a explicar cada una, como ayuda podrá acceder al tutorial de blitz3d, donde se explican en detalles estas instrucciones.

Introducción a los graficos 3d

Como sabremos los graficos 3d se crean a partir de poligonos, los cuales se hubican en un espacio 3d(El espacio esta compuesto por el eje x, eje y y eje z).

El poligono más común que podemos observar es el triangulo, apartir de este se pueden desarrollar modelos 3d más complejos.

La representación grafica de esta primitiva la podemos observar en la **Fig. 2**

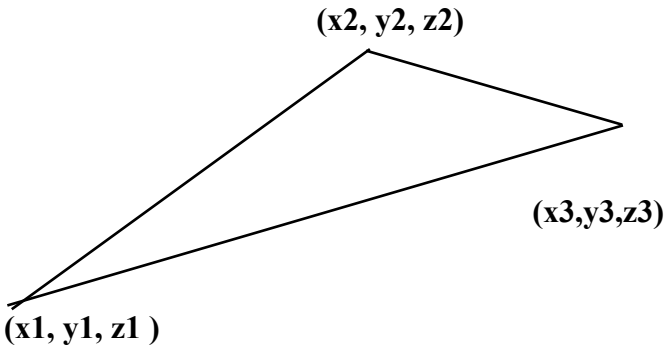

Fig. 2: Dibujo de un triangulo

Como vemos en la figura 2, el triangulo se encuentra compuesto por tres vertices:

(x1,y1,z1) - (x2,y2,z2) - (x3,y3,z3)

El vertice es El punto en común de los dos lados de un ángulo, es decir, el punto donde inicia un ángulo.

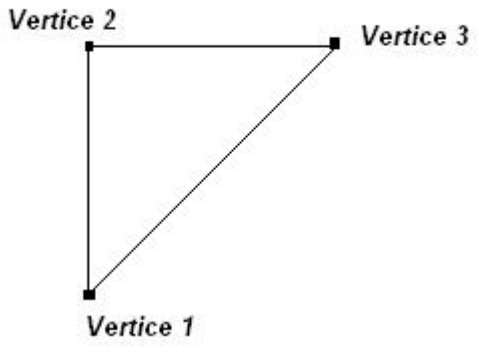

Fig. 3 . Ubicación de los vertices en un triangulo

Ejes cartesianos

Para facilitar la localización de un punto determinado utilizamos un sistema que se llama sistema de eje cartesiano. De forma arbitraria escogemos un punto como (0,0) y a partir de ahí empezamos a contar. El primer número se llama abscisa y es la coordenada que normalmente ponemos en sentido horizontal y que aumenta hacia la derecha (la X) y el segundo número se llama ordenada y es la coordenada que normalmente ponemos en sentido vertical y que aumenta hacia arriba (La Y).

Entonces, los ejes cartesianos serán los ejes X y eje Y en donde el eje X corresponde a la coordenada horizontal y el eje Y corresponde a la coordenada vertical.

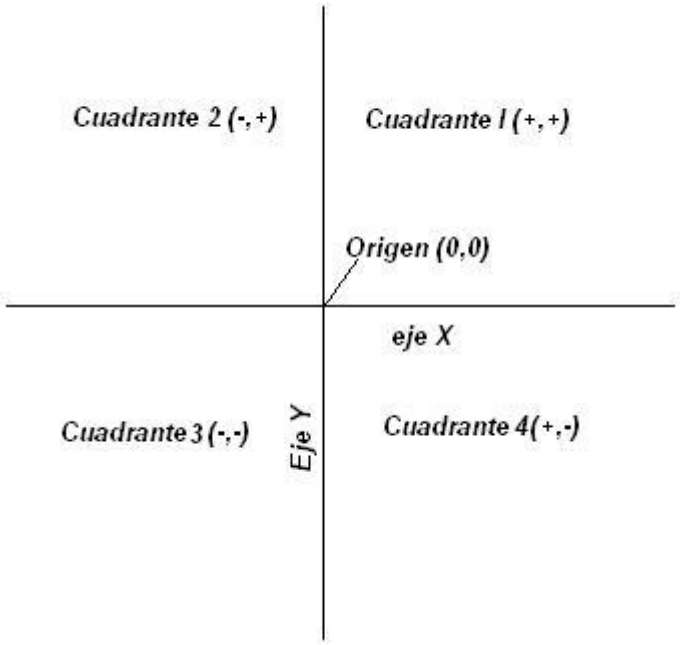

Fig. 4 Ejes cartesianos y sus componentes.

Los cuadrantes en un eje cartesiano corresponde a los valores que tomaran los puntos en un determinado cuadrante, por ejemplo:

(-3,8) estos valores corresponden al cuadrante 2 donde la x es negativa y la y es positiva.

Es muy importante aprender bien el uso de cuadrantes y de ejes cartesianos ya que para programar, por ejemplo el movimiento de una pelota, deberiamos saber que eje deberia aumentar para que la pelota se mueva.

Un ejemplo sería si queremos que una pelota se mueva de izquierda a derecha lo que deberiamos hacer es a la coordenada X de la pelota aumentarla

El pseudocodigo para realizar esta acción seria:

```
y= 60
mientra x<500
x = x+1
moverpelota(x,y)
fin mientras
```

con esto lograríamos que la pelota se mueva sobre el eje x a una altura fija de 60 puntos (este es el valor de Y)

Ejes cartesianos 3D

Para el espacio 3d a los 2 ejes anteriores se le debe agregar un tercer eje que daria la profundidad al gráfico

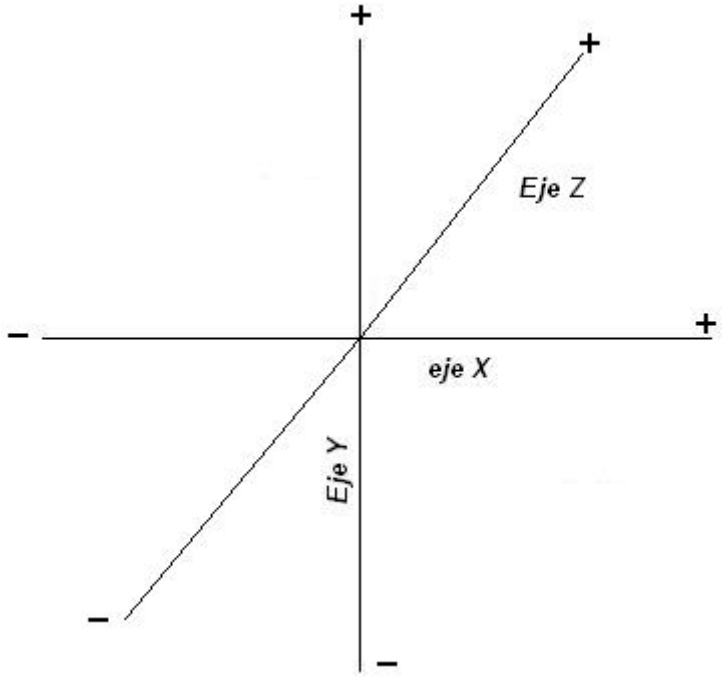

Fig. 5 Ejes cartesianos en un espacio 3D

Como vimos anteriormente para dibujar en un espacio 3d necesitaremos utilizar las tres cordenadas x,y,z.

En resumen, a la hora de programar juegos veremos que con los objetos que se dibujan en pantalla se pueden realizar diferentes acciones como por ejemplo moverlos. Para mover hacia el costado se trabaja con el eje x, para mover hacia arriba o abajo se trabaja con eje y, y en los juegos tridimensionales para ir hacia adelante o atras se utiliza el eje Z.

Como se dibuja un triangulo 3D en un programa de computación?

La forma antigua era:, dibujando pixel a pixel entre un punto del vertice y otro de esa forma se dibujaba la linea que unia los vertices, pero para hacer una simple figura geometrica se debia escribir mucho código.

Darkbasic cuenta con una función que crea un triangulo directamente, esta se llama:

MAKE OBJECT TRIANGLE

Los parametros que pide son, el número de primitiva y los 3 vertices.

Veamos un ejemplo de Esta función utilizando el programa Darkbasic:

Sintaxis

MAKE OBJECT TRIANGLE Número del Objeto, X1, Y1, Z1, X2, Y2, Z2, X3, Y3, Z3

Bien ahora construiremos un triangulo en Darkbasic,
escribamos el siguiente código en el ide:

```
sync on
sync rate 30
color Backdrop RGB(255,255,255)
ink rgb(0,0,0), 0

MAKE OBJECT TRIANGLE 1,0,0,0,0,10,0,10,10,0
MOVE OBJECT LEFT 1,5
MOVE OBJECT DOWN 1,5

SET OBJECT WIREFRAME 1,1
MOVE CAMERA -12

REPEAT
  SYNC
 UNTIL ESCAPEKEY() = 1
 END
```

Codigo1: creación de un Triangulo Simple

En el código 1 vemos como se puede contruir un triangulo, Ahora explicaremos el código

Sync on
lo que estamos indicando con esta instruccion es que el programa se encargará de refrescar la pantalla optimizando de esta forma los recursos para mostrar objetos. Por defecto Darkbasic tiene esta opción en **Off** Con lo cual el refresco de pantalla se realiza de forma automatica.

sync rate 30

con esta instrucción estamos diciendo que el refresco de pantalla se ejecutará a una taza de 30 frames por segundo.

color Backdrop RGB(255,255,255)

Con esta instrucción aplicamos un color de fondo, el sistema de colores utilizado es el RGB. Es de destacar en este caso se utilizo una instrucción compuesta, ya que RGB es otra instrucción la cual genera un valor entero que representa el color de la combinacion de los tres colores (Rojo, verde y Azul)

MAKE OBJECT TRIANGLE 1,0,0,0,0,10,0,10,10,0

Esta instrucción es la que estamos estudiando en estos momentos y lo que hace es construir un triangulo.

MOVE OBJECT LEFT 1,5
MOVE OBJECT DOWN 1,5

Mueve el objeto en una determinada posición, en este caso el primer parametro lo mueve a la izquierda y el segundo hacia abajo. El siguiente parametro indica el número del objeto, en este caso el triangulo y el ultimo el valor de avance, mientras mas grande más se mueve a la izquierda y abajo

SET OBJECT WIREFRAME 1,1

Lo que hace esta instrucción es colocar el objeto en modo wireframe con lo cual solo se dibujará las aristas del objeto. Esta función puede tomar 2 valores o 0 o 1(si es 0 mostrará el objeto en modo Solido)

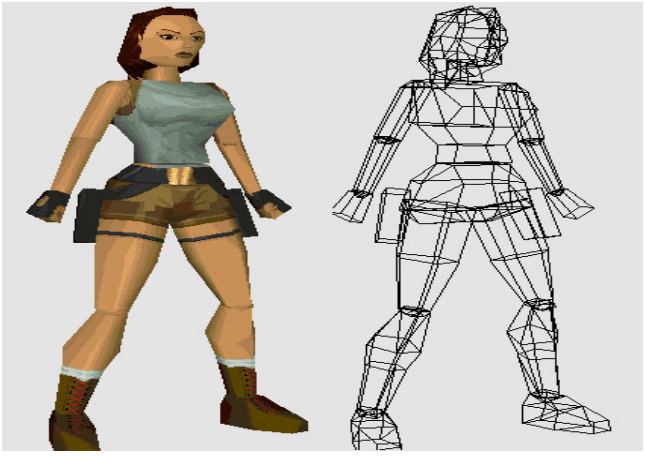

Fig. 6 Nuestra querida Lara en modo normal y WireFrame

MOVE CAMERA -12

Lo que hace este comando es colocar la camara a una distancia determinada. mientras el número de distancia sea mas pequeño, la camara se ubicará mas lejos.

Al utilizar este comando se puede usar de dos formas:

Sintaxis

MOVE CAMERA Valor Distancia
MOVE CAMERA Número de Camara, Valor Distancia

La diferencia es que el segundo sirve para cuando tenemos multiples camaras en nuestro juego.

```
REPEAT
  SYNC
 UNTIL ESCAPEKEY() = 1
```

Por último tenemos el bucle principal, el cual en este caso lo unico que realiza es el refresco de pantalla (**SYNC**) y espera a que el usuario presione la tecla ESC (**ESCAPEKEY())**

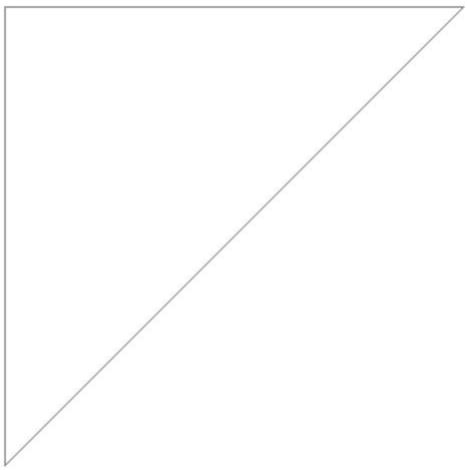

Fig7: El triangulo creado con el código 1

El la figura 7 observamos el resultado del codigo escrito anteriormente. Sería bueno que ahora el lector probara cambiando diferentes parametros del código para que puedan ver los resultados sobre el triangulo y de esta forma afianzar el conocimiento de cada instrucción.

Haciendo Girar El Triangulo

Lo proximo que haremos es dibujar un triangulo solido y hacerlo girar por la pantalla. Para realizar esto escribimos el siguiente código:

```
sync on
sync rate 100
HIDE MOUSE
color Backdrop RGB(0,40,40)
RANDOMIZE TIMER()
MAKE OBJECT TRIANGLE 1,-10,-10,0,-10,10,0,10,10,0
COLOR OBJECT 1, RGB(RND(255), RND(255),
RND(255))
MOVE CAMERA -20
REPEAT
  XROTATE OBJECT 1, OBJECT ANGLE X(1) + 1
  YROTATE OBJECT 1, OBJECT ANGLE Y (1) + 1
  SYNC
 UNTIL ESCAPEKEY() = 1

 SHOW MOUSE
 END
```

código 2: El codigo que hace girar nuestro triangulo.

Muy bien, ahora pasaremos a explicar los trozos de código que no fueron explicados anteriormente.

HIDE MOUSE

Esta instrucción lo que hace es ocultar el puntero del Mouse. para hacer que vuelva a aparecer utilizamos **SHOW MOUSE**

RANDOMIZE TIMER()

Randomize es un derivado de la funcion RND, la cual genera número aleatorios En este caso tendremos que usar lo que se conoce como “Semilla”, que es un número arbitrario usado para que empiece a funcionar el generador (Randomize). Si este número es el mismo siempre los números aleatorios devueltos serán siempre una misma serie, por lo tanto hay que usar como semilla un número lo más “aleatorio” posible como por ejemplo el número de segundos que han pasado a partir de las doce de la noche, cosa que nos devuelve la función TIMER. Para inicializar el generador de números aleatorios usaremos la instrucción RANDOMIZE seguida de la semilla, normalmente TIMER.

De esta forma cada vez que ejecutemos el programa tendremos un número distinto. El valor solo se repetiría si ejecutamos el programa otro día a la misma hora, mismo minuto, mismo segundo.

La instrucción RANDOMIZE TIMER debemos usarla una vez al principio de los programas que usen la función RND, pero no es necesario usarla más, de hecho repetirla en sitios como dentro de un bucle podría resultar contraproducente.

COLOR OBJECT 1, RGB(RND(255), RND(255), RND(255))

En esta función se le coloca un color al objeto 1 (en este caso el triangulo)

Su sintaxis es:

COLOR OBJECT número de objeto, número de color

Observemos que en nuestro caso estamos utilizando la función RGB, para que me devuelva el número de color.

Para obtener cada color, Rojo, verde y azul, Se utiliza la función RND la cual genera un número aleatorio entre 0 y el valor pasado por parametro en este caso el 255. es por esta función que anteriormente utilizamos **randomize** ya que de esta forma evitamos que siempre tome los mismos valores y por consiguiente muestre los mismos colores.

XROTATE OBJECT 1, OBJECT ANGLE X(1) + 1
YROTATE OBJECT 1, OBJECT ANGLE Y (1) + 1

Lo que hacen estas dos funciones es rotar el objeto 3d en el eje de cordenadas x e y, la sintaxis de esta instrucción es:

XROTATE OBJECT Número Objeto, XAngulo
YROTATE OBJECT Número Objeto, YAngulo

si observamos nuestro código, para darle el angulo de rotación al objeto 3d utilizamos **OBJECT ANGLE X(1) + 1 y OBJECT ANGLE Y (1) + 1.** El número entre parentesis representa el número de objeto, es decir que toma el angulo actual del objeto, luego le suma 1 para que comienze a girar.

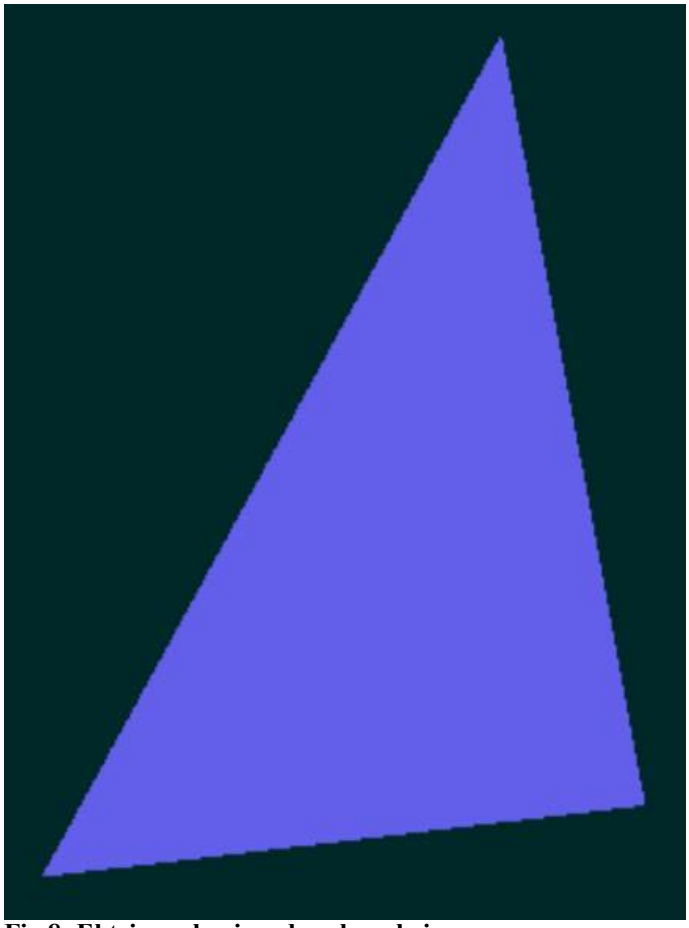

Fig 8: El triangulo girando sobre el eje x e y

Muy bien con todo lo que hemos visto tenemos para prácticar bastante, tengamos en cuenta que todo lo que en estos momentos aprendemos con un simple triangulo, en proximas lecciones nos servira para aplicarlo a modelos tridimensionales, por ejemplo de personajes para nuestro juego.

En el proximo número seguiremos trabajando con primitivas basicas, terminaremos de trabajar con el triangulo y empezaremos con las otras como por ejemplo el cubo, cilindro, etc.

Hasta el mes que viene

Alexis Jeansalle

Sobre el autor

Alexis Jeansalle, actualmente se encuentra estudiando la carrera de técnico en sistemas (con orientación al diseño multimedial y web). Trabaja como coordinador del área de capacitación en informática del gobierno de la provincia del chubut (Argentina). Es creador de la página www.academiaelearning.com.ar desde donde se podrán realizar cursos de informática totalmente gratis.