Kara Effector 3.2:
Effector Book
Vol. II [Tomo XXII]

Kara Effector 3.2:

El **Tomo XXII** es otro más dedicado a la librería **shape**, que como ya habrán notado, es la más extensa hasta ahora vista en el **Kara Effector**. El tamaño de esta librería nos da una idea de la importancia de las Shapes en un efecto karaoke, y es por ello que debemos tomarnos un tiempo en ver y conocer a cada una de las funciones y recursos disponibles para poder dominarlas.

Librería Shape [KE]:

shape.Pmove(x(s), y(s), dom, t1, t2, acc, off_t)

Esta función hace que el objeto karaoke se mueva siguiendo la trayectoria de la gráfica de la función definida por los parámetros **x(s)** y **y(s)** en el dominio **dom**.

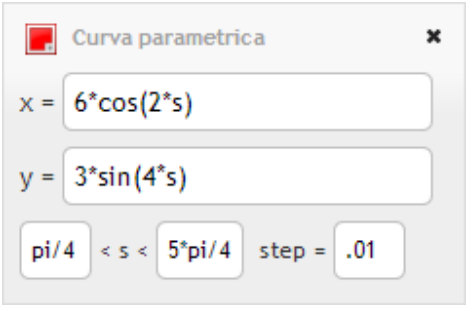
Los parámetros **t1** y **t2** son los tiempos de inicio y final del movimiento del objeto karaoke y sus valores por default son 0 y **fx.dur.**

El parámetro **dom** es el dominio de las funciones paramétricas **x(s)** y **y(s)**. Cuando es un valor numérico, el dominio será el intervalo cerrado entre 0 y dicho valor. Cuando es una **tabla**, el dominio va desde el valor del primer elemento hasta el segundo: {**dom_i**, **dom_f**}

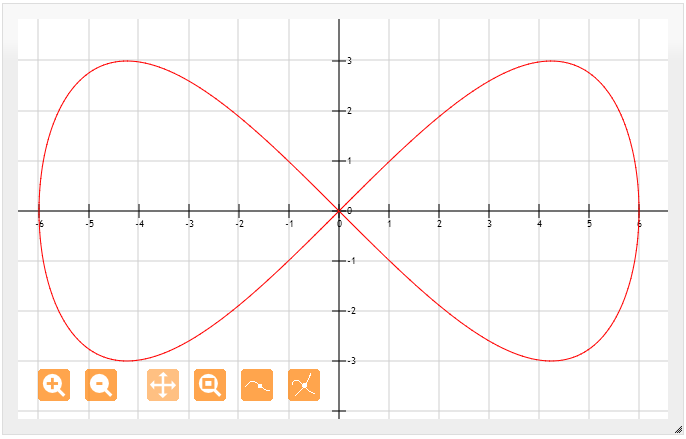
El parámetro **acc** es la aceleración del movimiento en las transformaciones que genera la función, y su valor por default es 1.

El parámetro **off_t** hace referencia al tiempo a añadir o restar de la duración de las transformaciones. Cada una de las transformaciones que genera esta función tiene una duración por default de 2.4***frame_dur**, y con **off_t** podemos añadir o quitar tiempo a esa duración.

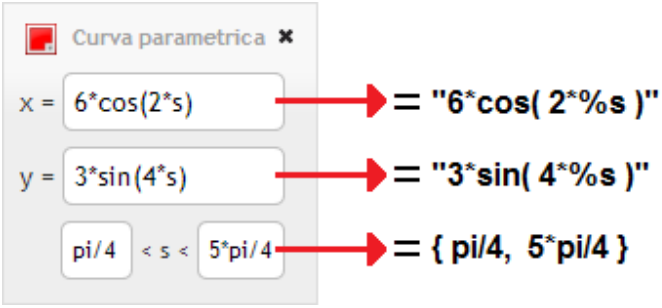
Ejemplo: www.fooplot.com



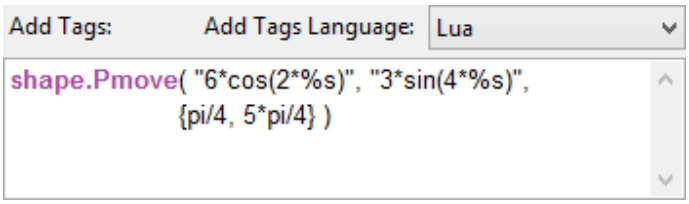
Esta es la gráfica de los anteriores parámetros:



Esto es lo que debemos hacer para pasar la información desde el graficador a la función:



- Poner las ecuaciones paramétricas entre comillas, ya sean simples y dobles.
- No olvidar poner siempre el símbolo del producto (*), o sea: 6*cos en vez de 6cos.
- Añadir el símbolo de porcentaje (%) antes de cada "s" que aparezca en las ecuaciones paramétricas.
- Hacer una **tabla** con los valores de inicio y final del dominio de las funciones paramétricas **x(s)** y **y(s).**



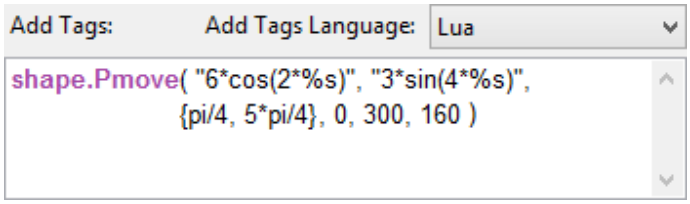
Entonces la función genera una shape invisible y una serie de transformaciones que harán que el objeto karaoke se mueva siguiendo la trayectoria de la gráfica generada por las dos ecuaciones paramétricas y el dominio asignado.

En el caso en que notemos que el movimiento sea poco perceptible, es porque las proporciones de las funciones son muy pequeñas. Las proporciones son los valores por el cual multiplicamos las funciones del anterior ejemplo:

- "cos(2*%s)" → "**6** * cos(2*%s)"
- "sin(4*%s)" → "**3** * sin(4*%s)"

Ejemplo:

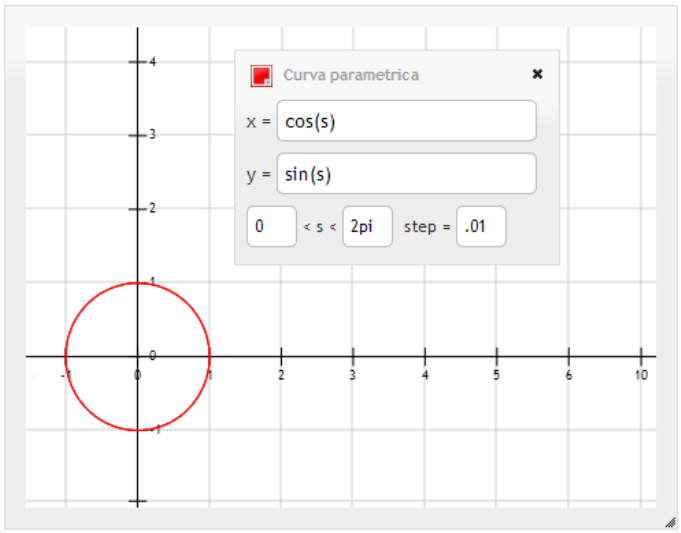
Este sería un ejemplo usando todos los parámetros de la función:



O sea que el movimiento empezará en 0 ms y terminará a los 300 ms. A parte de eso le estamos sumando 160 ms a la duración de las transformaciones generadas.

Ejemplo:

Los siguientes parámetros corresponden a la gráfica de un círculo de 1 px de radio, de modo que si los usamos para la función **shape.Pmove**, entonces no se notaría mucho el movimiento en el vídeo:

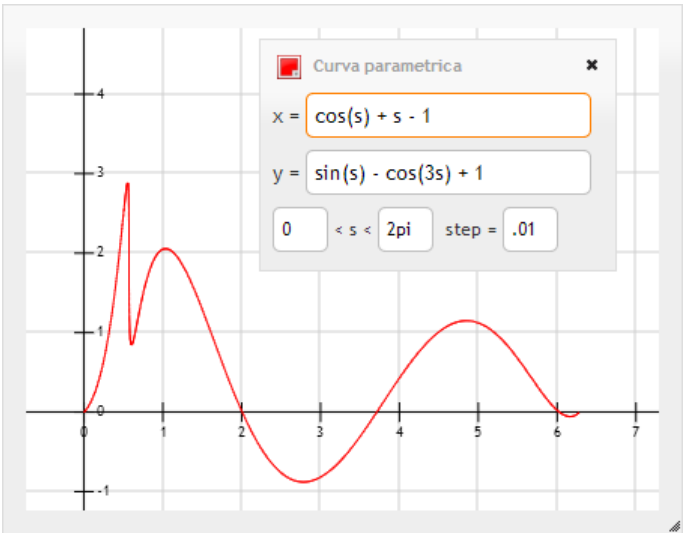


Lo que debemos hacer es multiplicar a cada ecuación paramétrica por un valor tal, que haga que el círculo sea más grande y así sea evidente el movimiento. Ejemplo:

- "100 * cos(%s)"
- "100 * sin (%s)"

Ejemplo:

Podemos definir las ecuaciones paramétricas en el graficador online y una vez que obtengamos una gráfica que sea de nuestro agrado, la podemos usar tal cual y luego ir modificando ambas proporciones con el fin de que el movimiento se ajuste a las necesidades de nuestro efecto:



Las ecuaciones para usar en la función quedarían:

- "cos(%s) + %s – 1"
- "sin(%s) – cos (3*%s) + 1"

Dominio:
- { 0, 2*pi }

Y para modificar las proporciones, debemos encerrar entre paréntesis a las ecuaciones paramétricas y así podremos multiplicar a cada una de ellas por el valor que necesitemos. Ejemplo:

- "45 * (cos(%s) + %s – 1)"
- "72 * (sin(%s) – cos (3*%s) + 1)"

shape.Smove(Shape, t1, t2, off)

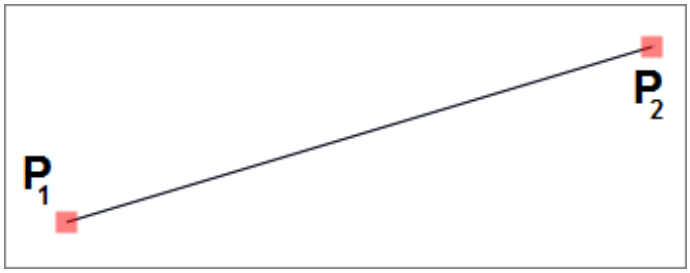
Esta función hace que el objeto karaoke se mueva siguiendo el contorno del perímetro de la shape ingresada "**Shape**", desde **t1** a **t2**.

Los parámetros **t1** y **t2** son los tiempos de inicio y final del movimiento del objeto karaoke y sus valores por default son 0 y **fx.dur.**

El parámetro **off** hace referencia al tiempo a añadir o restar de la duración de las transformaciones. Cada una de las transformaciones que genera esta función tiene una duración por default de 2***frame_dur**, y con **off** podemos añadir o quitar tiempo a esa duración.

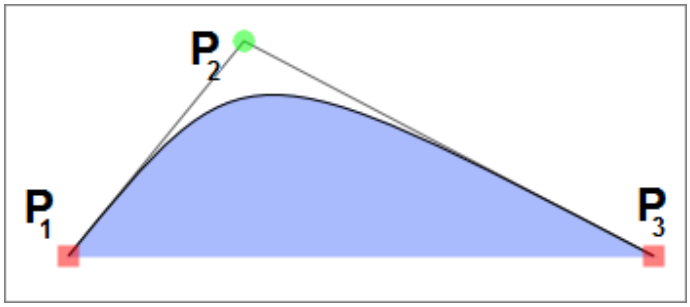
El parámetro **Shape** puede ser o una **tabla** o un **string**. Para el caso del **string**, debe ser el código de la **shape** en formato .**ass**, es decir el código que obtenemos de la **shape** en el **ASSDraw3**; o sea que podemos utilizar cualquiera de las Shapes que por default ya vienen en el **Kara Effector**. Y para el caso en que el parámetro **Shape** sea una **tabla**, ésta debe contener valores numéricos que cumplan con las siguientes características:

- Coordenadas de 2 puntos:
  Shape = { P_{x1}, P_{y1}, P_{x2}, P_{y2} }



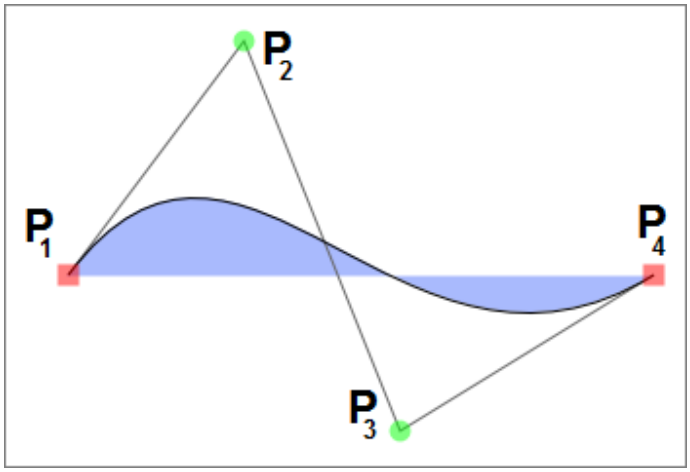
El objeto karaoke se moverá siguiendo la trayectoria de la línea recta que forman los dos puntos ingresados.

- Coordenadas de 3 puntos:
  Shape = { P_{x1}, P_{y1}, P_{x2}, P_{y2}, P_{x3}, P_{y3} }



El objeto karaoke se moverá siguiendo la trayectoria de la **Curva Bezier** que forman los tres puntos ingresados.

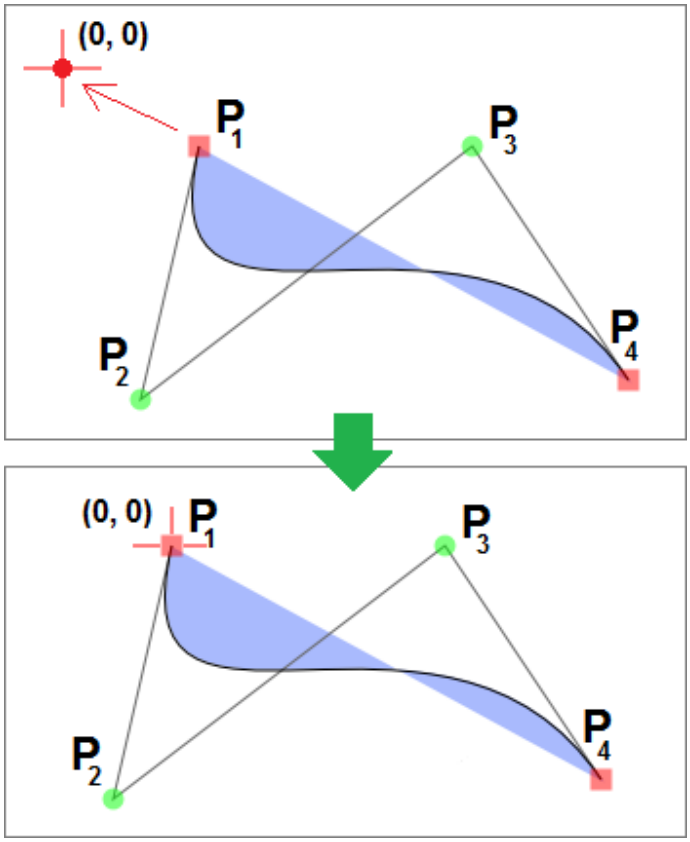
- Coordenadas de 4 puntos:
  Shape = { P_{x1}, P_{y1}, P_{x2}, P_{y2}, P_{x3}, P_{y3}, P_{x4}, P_{y4} }



El objeto karaoke se moverá siguiendo la trayectoria de la **Curva Bezier** que forman los cuatro puntos ingresados.

Es decir, que la función toma las coordenadas de los puntos ingresados y las convierte en una **shape**, para hacer que el objeto karaoke se mueva siguiendo dicha trayectoria.

Explicado esto, podemos decir que la función siempre toma como base a la trayectoria del perímetro de una **shape** para hacer que el objeto karaoke se mueva en el contorno de la misma. Lo siguiente que hace la función es mover la **shape** de tal forma que el primer punto quede en las coordenadas P = (0, 0) del plano en el **ASSDraw3**. Ejemplo:



Una vez se desplaza de la anterior forma a la shape, la función hace coincidir al centro del objeto karaoke con ese punto P = (0, 0), y ahí sí genera las transformaciones que hacen posible el movimiento.

Ejemplo:

Variables:

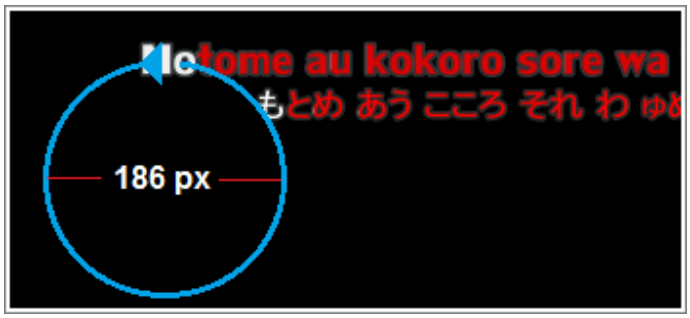
```
mi_shape = shape.size( shape.circle, 186 )
```

Declaramos una **shape** en la celda de texto "**Variable**", aunque sabemos que no es necesario, ya que lo podemos hacer directamente dentro de la función. La **shape** para este ejemplo es un círculo de 186 px de diámetro.

Add Tags: Add Tags Language: Lua

```
shape.Smove( mi_shape, fx.dur/2, fx.dur )
```

- **t1** = **fx.dur**/2
- **t2** = **fx.dur**



El objeto karaoke, en este caso las Sílabas, se moverán siguiendo como trayectoria al perímetro del círculo de 186 px de diámetro a partir de la mitad de la duración total de la línea de fx (**t1** = **fx.dur**/2), hasta el tiempo final de la misma, como se ve en la imagen anterior.

shape.Rmove(Dx, Dy, t1, t2, acc, off)

Esta función hace que el objeto karaoke se mueva aleatoriamente de forma lineal, desde su centro por default (**fx.move_x1**, **fx.move_y1**), sin exceder los límites **Dx** y **Dy**. El movimiento que se genera es perpetuo, dependiendo de los límites de tiempo **t1** y **t2**.

Los parámetros **t1** y **t2** son los tiempos de inicio y final del movimiento del objeto karaoke y sus valores por default son 0 y **fx.dur.**

El parámetro **acc** es la aceleración del movimiento en las transformaciones que genera la función, y su valor por default es 1.

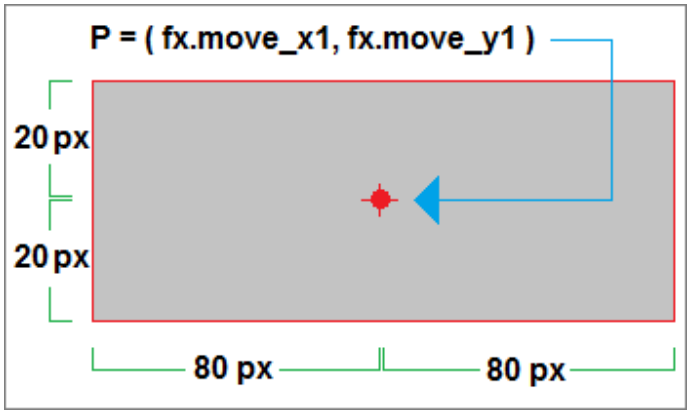
El parámetro **off** hace referencia al tiempo a añadir o restar de la duración de las transformaciones. Cada una de las transformaciones que genera esta función tiene una duración por default de 3.6***frame_dur**, y con **off** podemos añadir o quitar tiempo a esa duración.

Ejemplo:

- Dx = 80
- Dy = 20

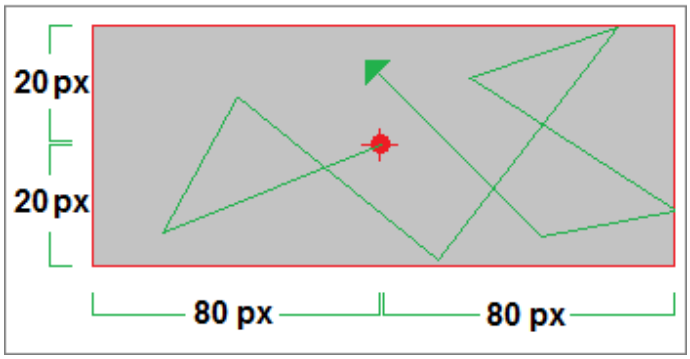
```
Add Tags:  Add Tags Language: Lua
shape.Rmove( 80, 20 )
```

Entonces la función crea un rectángulo imaginario con el doble de los valores ingresados en **Dx** y **Dy**, como ancho y alto de dicho rectángulo:



Y las transformaciones generadas harán que una **shape** invisible cree el **Movimiento Lineal Aleatorio** sin salirse nunca de los límites creados por el rectángulo imaginario:

Movimiento Lineal Aleatorio Delimitado



Esta función tiene una particularidad, que sin importar la cantidad de movimientos generados, la posición final será el mismo punto donde empezó, es decir que se moverá de forma aleatoria para finalmente terminar en donde estaba posicionado inicialmente: P = (**fx.move_x1**, **fx.move_y1**)

Ejemplo:

```
Add Tags:  Add Tags Language: Lua
shape.Rmove( 30, 25, 0, 400, 0.8 )
```

Entonces el objeto karaoke se moverá de forma aleatoria en un rectángulo imaginario de 60 X 50 px (el doble de cada valor ingresado en **Dx** y **Dy**), desde los 0 ms hasta los 400 ms y con una aceleración de 0.8; pero una vez transcurrido ese lapso de tiempo, volverá a su posición inicial de origen.

Las cuatro funciones que generan movimiento por medio de una shape invisible son:

- **shape.Lmove**
- **shape.Pmove**
- **shape.Smove**
- **shape.Rmove**

Aparte de **shape.Smove**, las otras tres funciones pueden generar movimiento con aceleración y cada una de ellas con las característica que hasta acá hemos visto.

El movimiento que generan las transformaciones de estas cuatro funciones es similar al generado por **shape.config**, con la diferencia que esta última genera un **loop** del objeto karaoke para dar la ilusión de movimiento.

Ya para terminar, el uso de estas cuatro funciones puede consumir una cantidad considerable de recursos de nuestra PC y volverla un poco lenta, lo que hará que no podamos apreciar en tiempo real a nuestro efecto. Todo dependerá del tipo de PC que cada uno tenga, pero pensando es este consumo de recursos, más adelante veremos otras cuatro funciones que también nos dan la posibilidad de generar movimiento acelerado con características similares a las que recientemente hemos visto y sin el efecto secundario de la lentitud de nuestra PC. Estas funciones pertenecen a la librería **tag** y en próximos **Tomos** las veremos en detalle.

Es todo por ahora para el **Tomo XXII**, pero la **librería shape** continuará en el próximo **Tomo**. Intenten poner en práctica todos los ejemplos vistos en este **Tomo** y no olviden descargar la última actualización disponible del **Kara Effector 3.2** y visitarnos en el **Blog Oficial**, lo mismo que en los canales de **YouTube** para descargar los nuevos Efectos o dejar algún comentario, exponer alguna duda o hacer alguna sugerencia. Pueden visitarnos y dejar su comentario en nuestra página de **Facebook**:

- www.karaeffector.blogspot.com
- www.facebook.com/karaeffector
- www.youtube.com/user/victor8607
- www.youtube.com/user/NatsuoDCE
- www.youtube.com/user/karalaura2012