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

Kara Effector 3.2:

El **Tomo XXIII** 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.

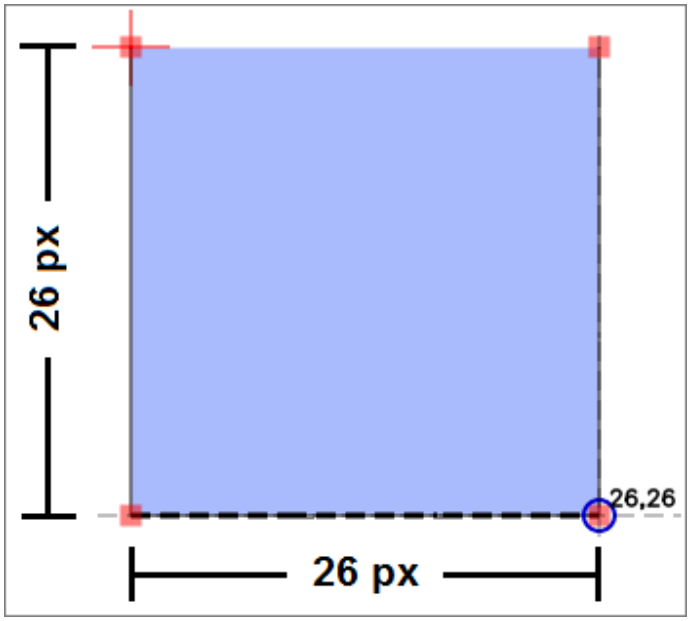
Librería Shape [KE]:

shape.length(Shape)

Esta función retorna la medida en pixeles de la longitud total del perímetro de la **shape** ingresada.

Ejemplo:

Shape.length(“m 0 0 l 0 26 l 26 26 l 26 0 l 0 0 “)

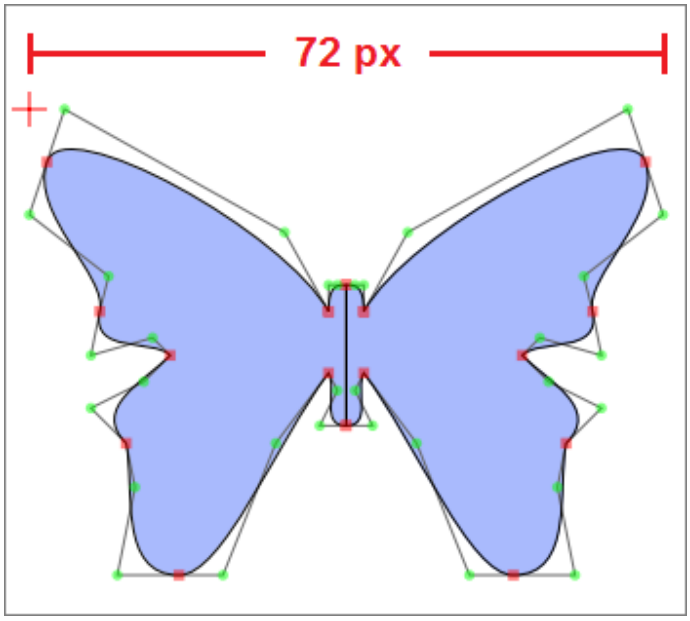


Entonces la función calculará la medida del perímetro de la **shape**, o sea: 26 + 26 + 26 + 26 = 104 px.

shape.width(Shape)

Esta función retorna la medida en pixeles del ancho total de la **shape** ingresada.

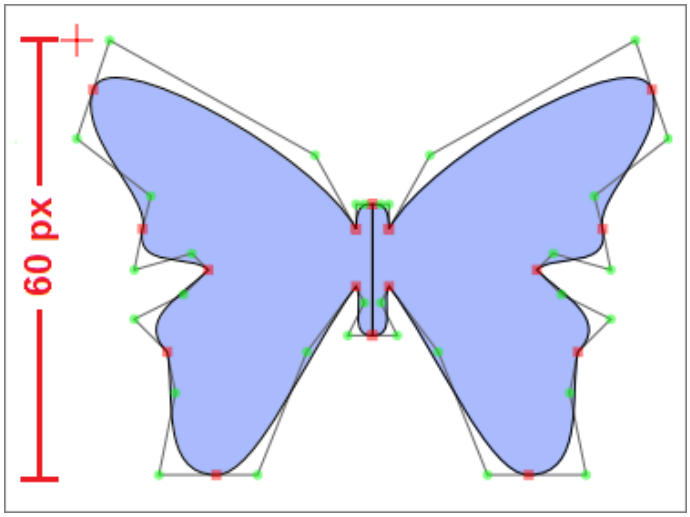
Ejemplo:



shape.height(Shape)

Esta función retorna la medida en pixeles de la altura total de la **shape** ingresada.

Ejemplo:



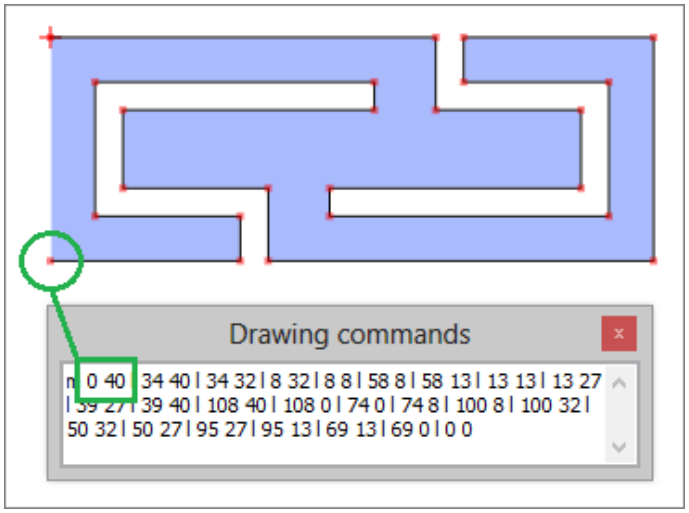
Las tres anteriores funciones están diseñadas para arrojar información básica de una **shape**, información como su longitud, su ancho y su altura. Esta información podrá ser usada en los efectos sin la necesidad de hacer los cálculos.

shape.firstpos(Shape, Px, Py)

Esta función es muy similar a **shape.displace** con la particularidad que mueve a todos los puntos de la **shape** ingresada teniendo en cuenta al primer punto de la misma, como referencia para hacerlo.

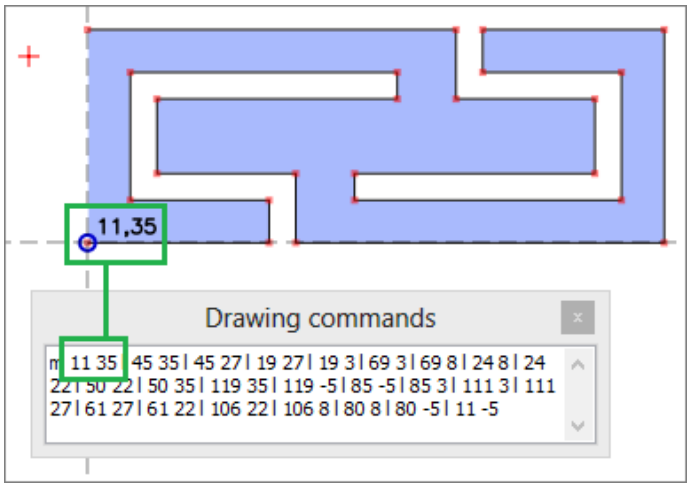
Ejemplo:

En la siguiente imagen vemos el primer punto de una shape, y con referencia a ese punto es que la función la moverá a la nueva posición que le indiquemos con los parámetros **Px** y **Py**:



Px es la coordenada respecto al eje “x” que tendrá el primer punto de la **shape**, y **Py** es la coordenada respecto al eje “y” de dicho punto:

shape.firstpos(mi_shape, 11, 35)



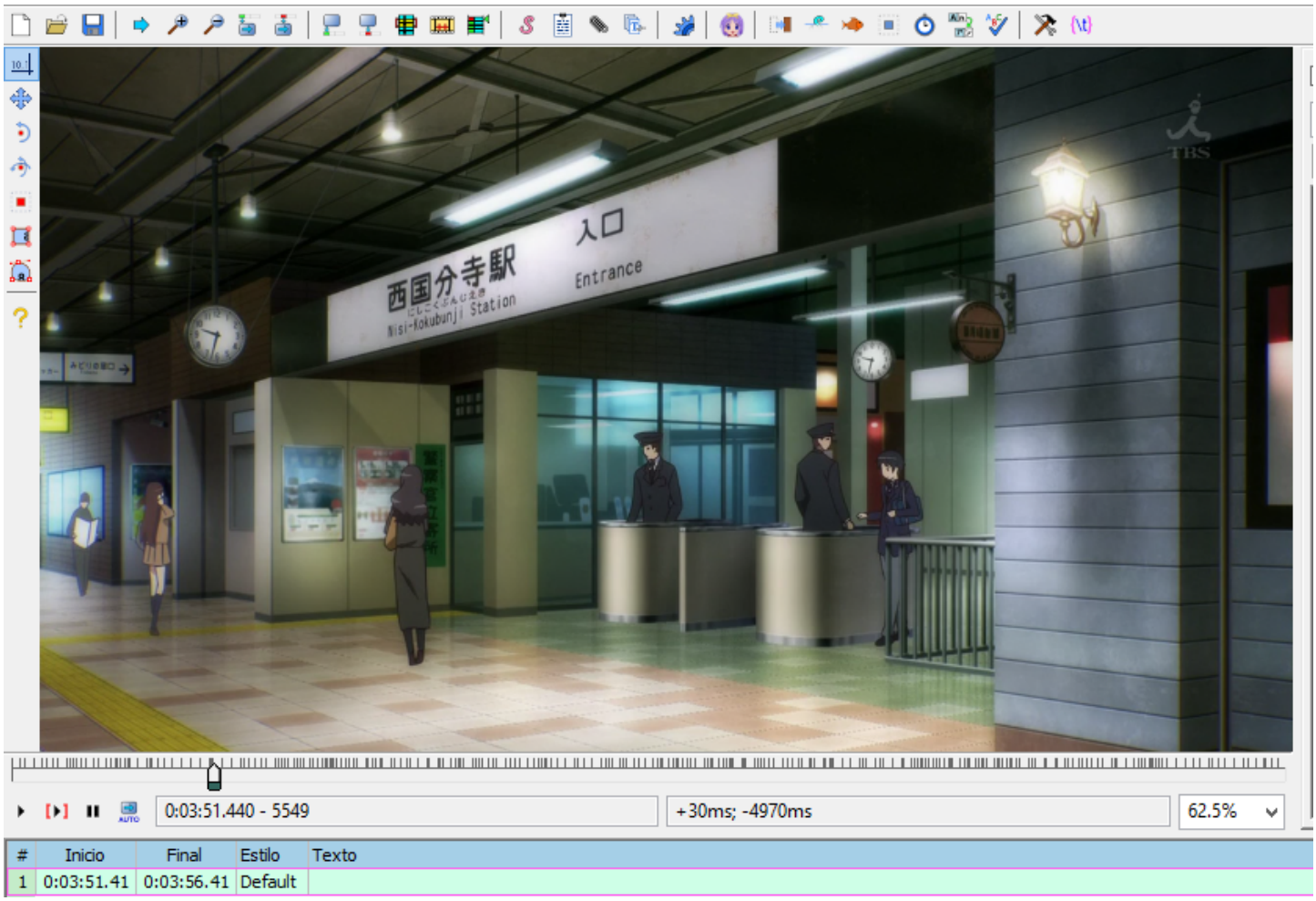
En conclusión, lo que hacemos al darles valores a los parámetros **Px** y **Py**, es asignar el punto exacto en donde quedará el primer punto de nuestra **shape** luego de ser desplazada. Esta función es ideal para reubicar la shape respecto al centro de la **Syl** y luego aplicar una de las funciones de **shape** de movimiento, como **shape.Smove**.

shape.from_clip( )

Esta función está pensada para ser usada en la edición, más que para hacer fx karaokes. Esta función ayuda a crear carteles y lo que hace es convertir un clip dibujado en la pantalla en un shape.

Ejemplo:

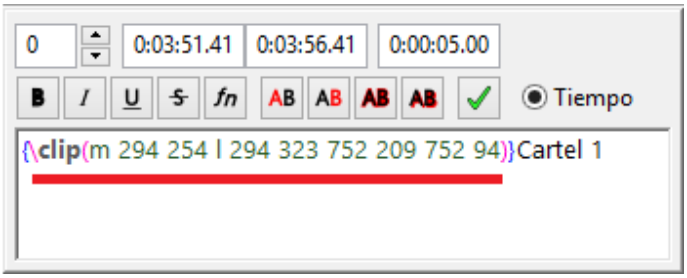
Ubicamos el frame en dónde está el cartel que queremos hacer y le damos los tiempos de inicio y final a una línea en la que posteriormente dibujaremos el clip:



En esa línea creada, escribimos algo, lo que sea, por ejemplo: “cartel 1” y seleccionamos la herramienta para recortar subtítulos en un área vectorial:

Y dibujamos el contorno del cartel:

Hecho esto, en nuestra línea que previamente habíamos creado, se debe ver el código del clip, algo como esto:



Y por último, en **Return [fx]** llamamos a la función:

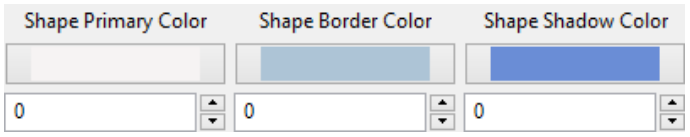
Return [fx]:

```
shape.from_clip( )
```

Y al aplicar ya podemos ver la **shape** justo en el lugar en donde está el cartel:

#	Inicio	Final	Estilo	Texto
1	0:03:51.41	0:03:56.41	Default	✳Cartel 1
2	0:03:51.41	0:03:56.41	Default	✳m 294 254 l 294 323 l 752 209 l 752 94

Recordemos que los colores y transparencias de una shape los podemos modificar con las herramientas de la **Ventana de Modificación** del **KE** destinadas para ello:



shape.redraw(Shape, tract)

Esta función cumple con la tarea de redibujar la shape que ingresemos, pero dividiendo a las partes que la componen en tramos muchos más cortos.

El parámetro **tract** es la medida en pixeles en la que será dividido cada segmento de la **shape** y debe ser un número mayor a cero. Su valor por default es 2.

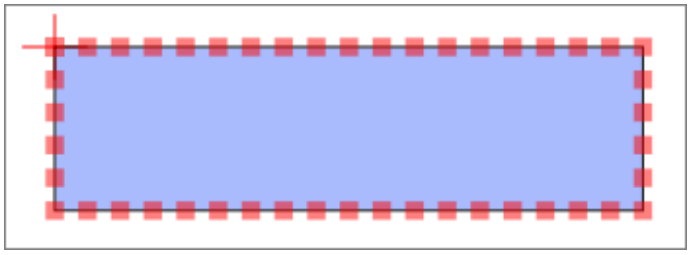
Ejemplo:

Entonces ponemos el código de la anterior **shape** dentro de la función y aplicamos:

Return [fx]:

```
shape.redraw( "m 0 0 l 0 10 l 38 10 l 38 0 l 0 0 " )
```

Y obtendremos esto:

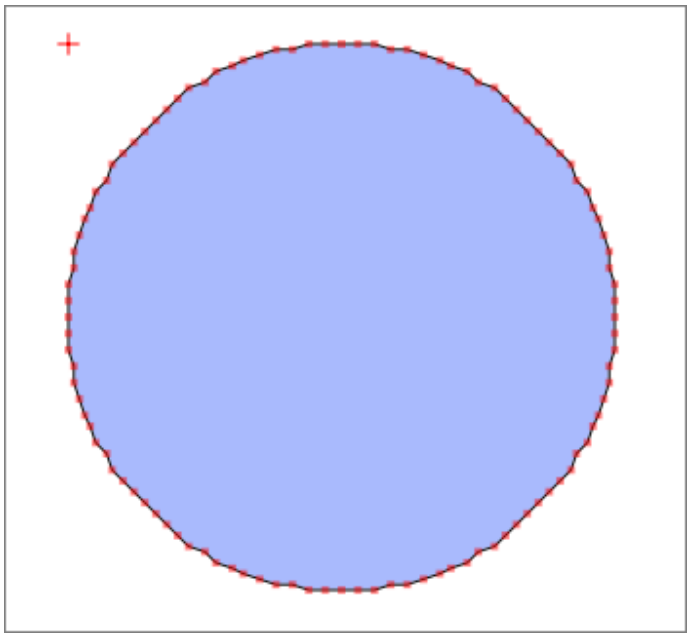


La **shape** que genera la función es la misma, pero dibujada con segmentos cortos de 2 pixeles de largo, ya que no pusimos el parámetro **tract**, entonces la función lo asume por default, es decir: 2 px.

Ejemplo:

Return [fx]:

```
shape.redraw( shape.circle, 3 )
```



Entonces la función redibuja el círculo con tramos rectos de 3 pixeles de longitud.

Puesto en práctica estos dos ejemplos, la pregunta sería: ¿por qué redibujar una **shape** en segmentos rectos más cortos?

Una de las varias respuestas a esa pregunta la puede responder la siguiente función de la **librería shape**:

```
shape.modify( Shape, modify )
```

Esta función modifica a los puntos de una shape por medio de una función ingresada en el parámetro **modify**, que debe tener la siguiente estructura:

modify:

```
function( x, y )
    -- Instrucciones --
    return  format( "%d %d", x, y)
end
```

Ejemplo:

Aplicamos el **shape.redraw** al círculo y la definimos como una variable en la celda de texto "**Variables**":

Variables:

```
mi_shape = shape.redraw( shape.circle, 1 )
```

Seguido de esto, definimos la función que modificará a la anterior **shape**:

Variables:

```
mi_shape = shape.redraw( shape.circle, 1 );
mi_modify = function( x, y )
        x = x + R(-6,6)        Instrucciones
        return format( "%d %d", x, y)
end
```

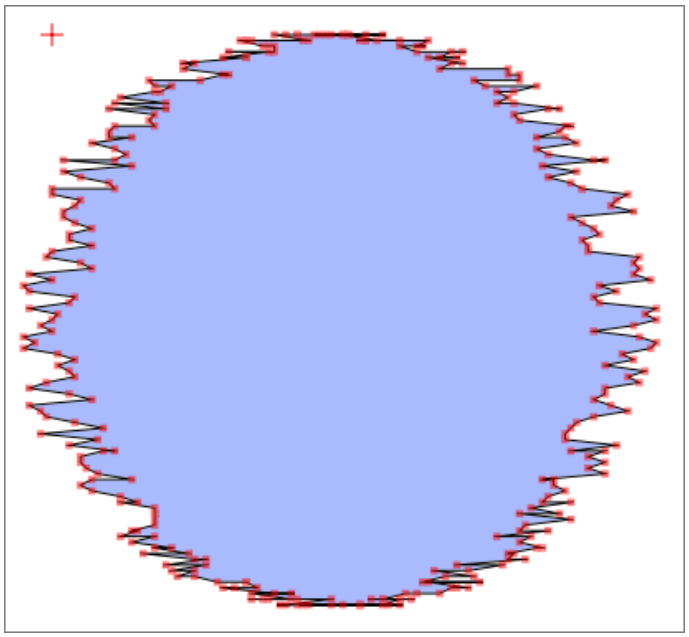
La instrucción de la anterior función declarada es que tome las coordenadas respecto al eje "x" de la **shape** y las modifique en una cantidad aleatoria entre -6 y 6 px.

Ahora llamamos a la función **shape.modify** con las dos variables que acabamos que declarar:

Return [fx]:

```
shape.modify( mi_shape, mi_modify )
```

Lo que dará por resultado algo como esto:



Y lo que antes era un simple círculo, ahora es uno que está distorsionado, como la estática de una TV sin señal.

Otra característica de la función **shape.modify** es que contiene internamente a la función **shape.info**, que como recordaremos, nos da la siguiente información de la **shape**:

- **minx** ← mínima coordenada en "x"
- **maxx** ← máxima coordenada en "x"
- **miny** ← mínima coordenada en "y"
- **maxy** ← máxima coordenada en "y"
- **w_shape** ← ancho de la **shape**
- **h_shape** ← alto de la **shape**

Y esta información la podemos usar dentro de la función que usaremos para modificar a la **shape**. Ejemplo:

```
1  mi_modify = function( x, y )
2      mod = (y - miny)/h_shape
3      x = x + 20*sin( 2*pi*mod )
4      y = 2*y
5      return format( "%d %d", x, y )
6  end
```

Generalmente uso el programa **Notepad++** para hacer las funciones, pero no es obligatorio hacerlo, ya que se pueden hacer directamente en el **Kara Effector** o hasta usar el Bloc de Notas es suficiente para ello.

La anterior función de modificación, declarada en la celda de texto "**Variable**" se vería así:

Variables:

```
mi_shape = shape.redraw( shape.circle, 1 );
mi_modify = function( x, y )
        mod = (y - miny)/h_shape
        x = x + 20*sin( 2*pi*mod )
        y = 2*y
        return format( "%d %d", x, y )
end
```

Lo que quieren decir las instrucciones de la anterior función es que a cada coordenada "x" de la **shape** ingresada, se le sume esta función trigonométrica:

20*math.sin(s) → 0 <= s <= 2*pi

Y que a cada coordenada en "y" se multiplique por 2:

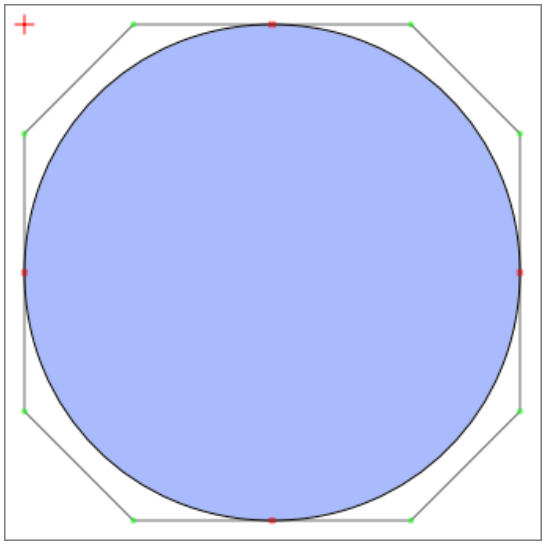
y = 2*y

Y ya sabemos lo que debemos poner en **Return [fx]:**

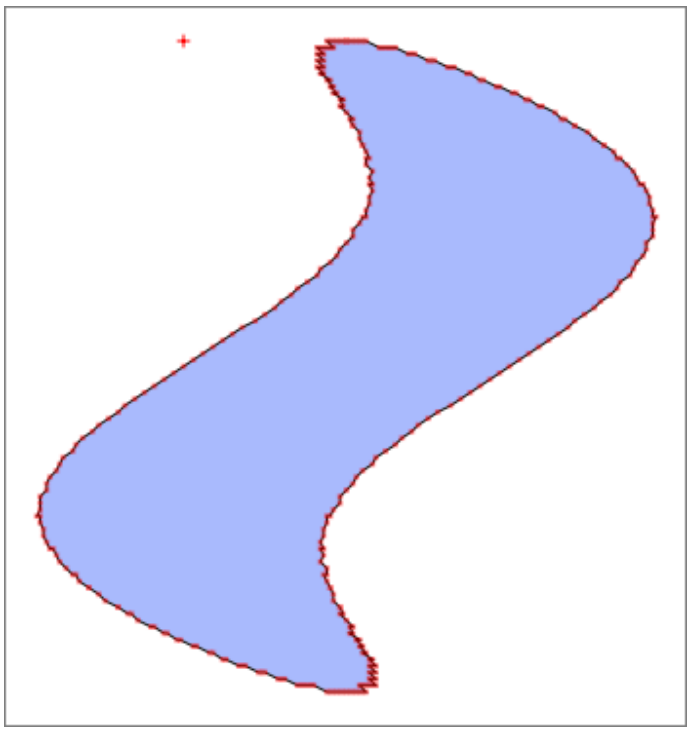
Return [fx]:

```
shape.modify( mi_shape, mi_modify )
```

Y convertiremos esta **shape**:



En esto:



Imaginen hacerle estas deformaciones al texto, sería algo genial para hacer nuestros fx. Pues es posible hacerlo y más adelante les mostraremos la forma de lograrlo, pero primero es importante que se vayan familiarizando con las funciones necesarias para ello.

Es todo por ahora para el **Tomo XXIII**, 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/NatsuoKE
- www.youtube.com/user/karalaura2012