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

Kara Effector 3.2:

En este **Tomo XXV** veremos una función más de la **librería shape** y haremos una corta pausa de la misma, ya que consideramos más importante ver otras funciones más de otras librerías que aún nos restan por ver, pero no se preocupen, hasta la fecha solo restan 3 funciones más de la **librería shape**, por documentar y ya llegará el momento de verlas con más detenimiento.

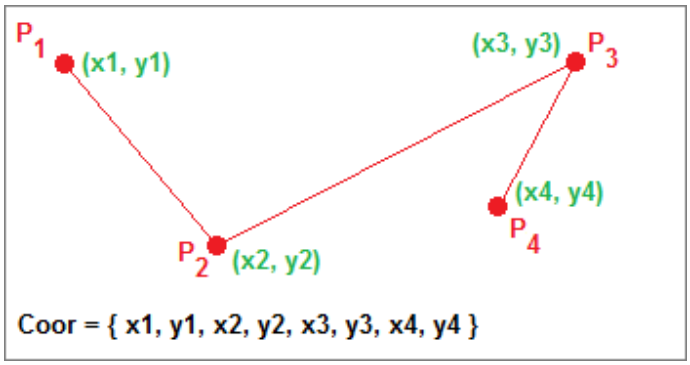
La nueva librería que empezaremos a ver a partir de este **Tomo XXV** es la **librería text**, que contiene una serie de funciones interesantes que sé que le sacarán el mayor provecho en sus proyectos.

Librería Shape [KE]:

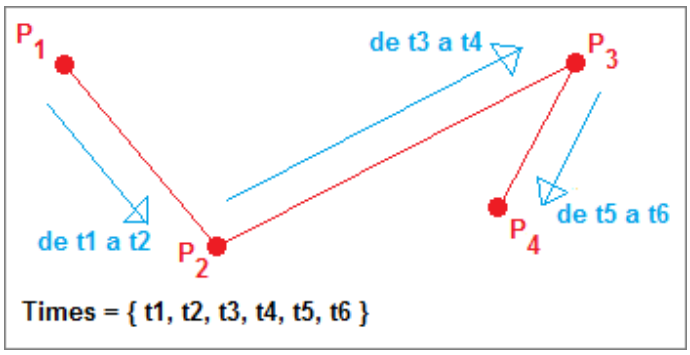
shape.Lmove2(Coor, Times)

Esta función es similar a **shape.Lmove**, pero con la diferencia que necesita dos tablas como parámetros para llevar a cabo su labor.

El parámetro **Coor** es la **tabla** de las coordenadas de los puntos en los que se desplazará nuestro objeto karaoke:



El parámetro **Times** es la **tabla** de los tiempos iniciales y finales de los movimientos entre punto y punto de la **tabla** del parámetro **Coor**:



Ejemplo:

Este será un sencillo ejemplo de su aplicación y puede ser usado en las líneas de traducción, de manera que usaremos el efecto con **Template Type: Translation Line**

Primero definimos nuestras dos **tablas** en la celda de texto "**Variables**", aunque este procedimiento no es del todo necesario, ya que podemos ingresar directamente las tablas en la función sin tener de definirlas, pero lo hago por simple organización:

Variables:

```
puntos = { line.center - 500, line.middle,
           line.center, line.middle,
           line.center + 500, line.middle };
tiempos = { 0, 200, line.dur - 200, line.dur}
```

Entonces tenemos:

- **P1 = line.center – 500, line.middle**
- **P2 = line.center, line.middle**
- **P3 = line.center + 500, line.middle**
- **t1 = 0**
- **t2 = 200**
- **t3 = line.dur – 200**
- **t4 = line.dur**

Y la función hará la siguiente:

- Moverá la línea de texto de P1 a P2 desde t1 a t2, o sea, desde los 0 ms hasta 200 ms.
- Desde t2 a t3, la línea de texto permanecerá estática, o sea que la línea no se moverá desde los 200 ms hasta line.dur – 200 ms.
- Moverá la línea de texto de P2 a P3 desde t3 a t4,o sea, desde line.dur – 200 ms hasta line.dur

Y para ello ponemos en **Add Tags** así:

Add Tags Language: Lua

```
shape.Lmove2( puntos, tiempos )
```

La cantidad de puntos que podemos ingresar en la función **shape.Lmove2** también es ilimitada, lo que conlleva a una cantidad ilimitada de movimientos lineales. Las aplicaciones son múltiples y todo dependerá de la imaginación.

La ventaja de esta función es que podemos hacer la cantidad de movimientos lineales que deseemos, y no necesariamente deben ser consecutivos ni durar la misma cantidad de tiempo, ya que controlamos los tiempos de cada desplazamiento, así como los lugares a los que se moverá.

Librería Text [KE]:

Esta librería, como su nombre lo indica, está enfocada en el texto de nuestros proyectos. Esta librería contiene una serie de funciones destinadas a la modificación, transformación y mejoramientos de los recursos de las líneas de texto.

text.upper(Text)

Esta función convierte el texto que le ingresemos en el parámetro **Text**, a mayúsculas.

El parámetro **Text** debe ser un string de texto, y su valor por default dependerá del **Template Type** seleccionado en el efecto que vayamos a aplicar:

Template Type [fx]	Texto por Default
Syl	
Line	line.text_stripped
Word	word.text
Syl	syl.text
Furi	furi.text
Char	char.text
Convert to Hiragana	hira.text
Convert to Katakana	kata.text
Convert to Romaji	roma.text
Translation Line	line.text_stripped
Translation Word	word.text
Translation Char	char.text
Template Line [Word]	line.text_stripped
Template Line [Syl]	line.text_stripped
Template Line [Char]	line.text_stripped

Ejemplo:

Template Type [fx]: Translation Line

Return [fx]:

```
text. upper( )
```

line.text_stripped por default

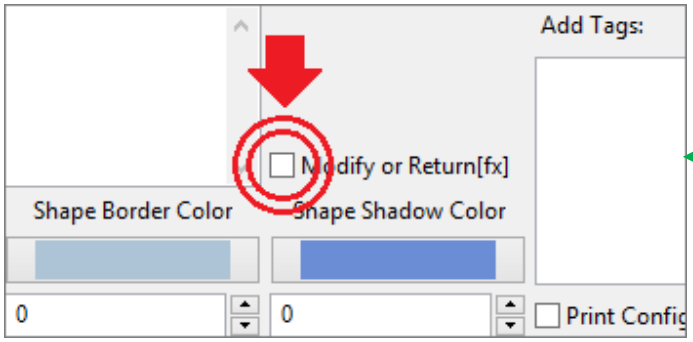
Y acá notamos cómo el texto se cambió todo a mayúsculas:

	Siento los pensamientos que dejaste atrás
	Me aferraré a ti y nunca te soltaré
	Dos corazones que se buscan conforman este sueño
[Fx]	✱MIS MEJILLAS SE MANCHAN CON LÁGRIMAS DE SOLEDAD
[Fx]	✱PERO PUEDO SENTIR LA LLEGADA DEL AMANECER
[Fx]	✱QUE ME ATRAE CON RUMBO HACIA LOS CIELOS
[Fx]	✱LA ESPERANZA ESPERA AL OTRO LADO, POR ESO VOLARÉ
[Fx]	✱ME PIERDO EN EL CAMINO CADA VEZ QUE TE BUSCO
[Fx]	✱SIENTO LOS PENSAMIENTOS QUE DEJASTE ATRÁS

Si ponemos algo distinto en el parámetro **Text**, la función lo transformará automáticamente en mayúsculas. Ejemplo:

- **text.upper("texto demo") → "TEXTO DEMO"**
- **text.upper("Ejemplo n° 2") → "EJEMPLO N° 2"**
- **text.upper("Aegisub") → "AEGISUB"**

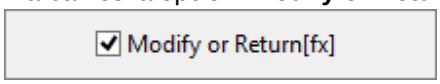
Otra forma de usar esta función es apoyándonos de esta herramienta del **Kara Effector**:



La opción "**Modify or Return[fx]**", al estar marcada, hace que el efecto en vez de generar nuevas líneas de fx, modifique a las líneas de nuestro script. Como su nombre lo indica, con esta opción decidimos si queremos modificar a nuestras líneas del archivo .ass o queremos generar un efecto a partir de líneas nuevas de fx.

Ejemplo:

- **Template Type**: Line o Translation Line
- Marcamos la opción "**Modify or Return[fx]**"



- En **Return [fx]** llamamos a nuestra función:

Y lo que sucederá es que no se generarán líneas de fx, sino que las líneas seleccionadas para que se le aplicará el efecto, se pasarán a mayúsculas:

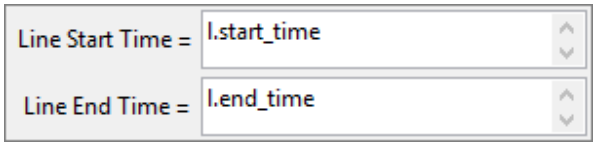
Estilo	Texto
Hiragana	✱す✱れ✱ち✱が✱う ✱い✱し✱き ✱て ✱が ✱ふ✱れ✱た ✱ょ ✱ね
Hiragana	✱つ✱か✱ま✱え✱る ✱ょ ✱し✱っ✱か✱り
Hiragana	✱も✱と✱め ✱あ✱う ✱こ✱こ✱ろ ✱そ✱れ ✱わ ✱ゅ✱め ✱の ✱あ
English	MIS MEJILLAS SE MANCHAN CON LÁGRIMAS DE SOLEDAD
English	PERO PUEDO SENTIR LA LLEGADA DEL AMANECER
English	QUE ME ATRAE CON RUMBO HACIA LOS CIELOS
English	LA ESPERANZA ESPERA AL OTRO LADO, POR ESO VOLARÉ
English	ME PIERDO EN EL CAMINO CADA VEZ QUE TE BUSCO
English	SIENTO LOS PENSAMIENTOS QUE DEJASTE ATRÁS
English	ME AFERRARÉ A TI Y NUNCA TE SOLTARÉ
English	DOS CORAZONES QUE SE BUSCAN CONFORMAN ESTE SUEÑ

Modify or Return [fx]

Esta opción del **Kara Effector** nos da la posibilidad de decidir si queremos modificar las líneas de nuestro script .ass o si queremos generar nuevas líneas fx.

Si marcamos esta opción, podemos modificar a nuestras líneas del script .ass de dos formas diferentes.

1. **Su contenido**:
   El contenido de las líneas se modificará solo con todo aquello que pongamos en la celda de texto **Return [fx].**
2. **Sus tiempos**:
   Podemos modificar los tiempos de inicio y final de las líneas de nuestro script .ass con las celdas de texto del **Kara Effector** destinadas para ello:



   Cualquier tiempo que añadamos o sustraigamos en estas dos celdas, modificarán los tiempos de las líneas del script .ass

Con esta opción marcada, solo funcionan estas tres celdas de texto de la **Ventana de Modificación** del **Kara Effector**, es decir que el resto de las herramientas en esta Ventana quedan inhabilitadas para modificar a la líneas de nuestro script .ass

text.lower(Text)

Es la función opuesta a **text.upper**, y convierte el string de texto ingresada en ella, a minúsculas.

Ejemplo:

- **text.lower("Texto Demo") → "texto demo"**
- **text.lower("EJEMPLO N° 2") → "ejemplo n° 2"**
- **text.lower("AEGISUB") → "aegisub"**

text.infx(select_fx)

Esta función aplica un efecto total o parcialmente, únicamente a las Sílabas previamente marcadas en nuestro script .ass

Hay dos formas diferentes con las que podemos marcar las Sílabas del script:

- **-fx**
- **+fx**

Estas marcas deben ir dentro de los tags de las Silabas.

La marca (**-fx**) selecciona únicamente a la sílaba que la contenga en su tag. Ejemplo

{\k19}ma{\k18-fx}yo{\k28}i {\k32}na{\k31-fx}ga{\k35-fx}ra

Sílabas marcadas:

- "yo"
- "ga"
- "ra"

La marca (**+fx**) selecciona a todas las Sílabas desde la marcada hasta el final de la línea karaoke, o hasta la siguiente marca (-fx) que se encuentre a su derecha:

{\k19}ma{\k18+fx}yo{\k28}i {\k32}na{\k31}ga{\k35}ra

De este modo, la marca (**+fx**) seleccionará a todas las Sílabas desde "yo" hasta la última, o sea la Sílaba "ra".

{\k19}ma{\k18+fx}yo{\k28}i {\k32-fx}na{\k31}ga{\k35}ra

Hecho así, quedan seleccionadas las Sílabas desde "yo" hasta "na".

{\k19}ma{\k18+fx}yo{\k28}i {\k32-fx}na{\k31}ga{\k35-fx }ra

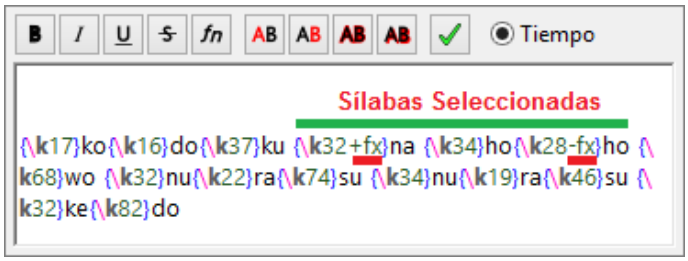
Y así, quedan seleccionadas las Sílabas desde "yo" hasta "na" y la Sílaba "ra".

Una vez entendida la forma en las que podemos marcar las Sílabas de las líneas karaoke, veremos varios ejemplos de cómo usar la función **text.infx**, ya que tiene tres diferentes modos de uso.

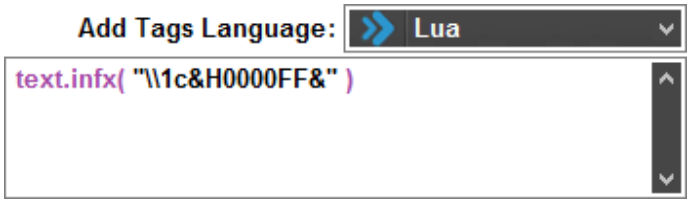
- **Modo 1**: aplicar un efecto parcial. Un **string**:

Ejemplo:

Marcamos algunas sílabas, no importa de cuál o de cuántas líneas karaokes:



Y en **Add Tags** agregamos algún **string** en la función:

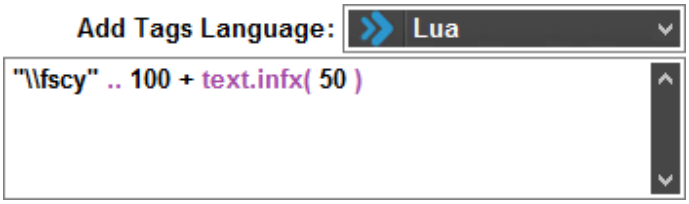


Entonces al aplicar el efecto, la función solo agregará el string ingresado únicamente a las Sílabas seleccionadas previamente:

Notamos cómo el string (en este caso el color primario rojo) fue agregado solo a las Sílabas marcadas, a las demás no le agregó absolutamente nada.

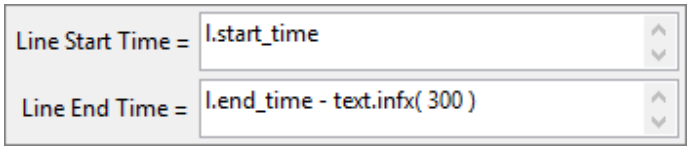
- **Modo 2**: aplicar un efecto parcial. Un **número**.

Ejemplo:



Entonces la función sumará los 50 a los 100 que están antes de ella, solo a las Sílabas marcadas, a las restantes le sumará 0. O sea que la función, a las Sílabas que no están marcadas les agregará **"\\fscy100"** y a las que sí lo están, les agregará **"\\fscy150"**:

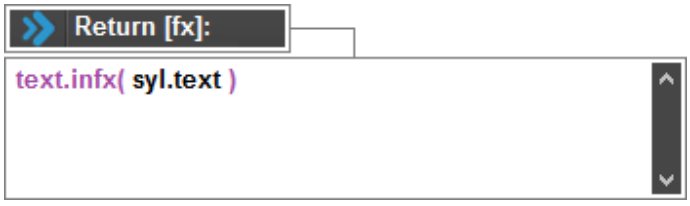
Con este modo de adicionar o sustraer un número con la función **text.infx**, también lo podemos hacer en las celdas de texto de tiempo. Ejemplo:



- **Modo 3**: aplicar un efecto total. Únicamente a las Sílabas marcadas:

Ejemplo:

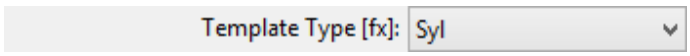
Para que un efecto se aplique únicamente a las Sílabas que previamente hemos marcado, lo que debemos hacer es usar la función en la calda de texto **Return** [fx], así:



Entonces el efecto que hayamos elegido solo se aplicará a la Sílabas marcadas, el resto no se tendrán en cuenta, y por ende no saldrán en pantalla:

En resumen, la función usada en cualquier celda de texto que no sea **Return [fx]** aplica el efecto de modo parcial a las sílabas marcadas, al resto solo se le aplicará el efecto general y no el que esté dentro de la función. Y al usar la función en **Return [fx]** como en el último ejemplo, el efecto general solo se aplicará a las Sílabas marcas y las demás no se tendrán en cuenta.

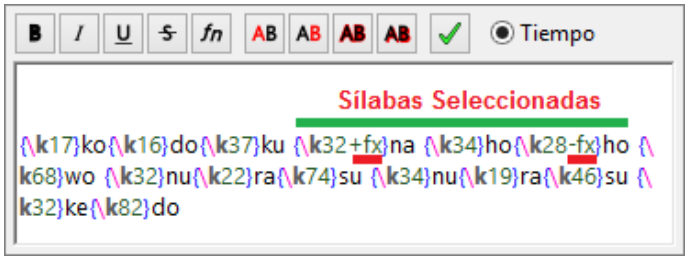
Solo resta decir que esta función, dado que marcamos las Sílabas, está diseñada únicamente para ser usada en el modo **Template Type: Syl**



text.outfx(select_fx)

Esta es la función opuesta a **text.infx**, es decir que nunca toma en cuenta a las Sílabas marcadas con (**+fx**) o con (**-fx**).

Ejemplo:



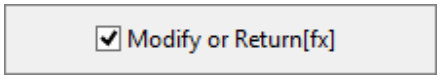
Return [fx]:

```
text.outfx( syl.text )
```

Y vemos el resultado opuesto de la función **text.infx**, es decir que no toma en cuenta a las Sílabas marcadas:

text.kara( )

Esta función está diseñada para ser usada con la opción "**Modify or Return [fx]**" marcada, y dentro de la celda de texto **Return [fx]**.



Return [fx]:

```
text.kara( )
```

Y lo que hace esta función es convertir las líneas normales de nuestro script .ass a líneas karaoke:

Ejemplo:

Convierte estas líneas:

English	Mis mejillas se manchan con lágrimas de soledad
English	Pero puedo sentir la llegada del amanecer
English	Que me atrae con rumbo hacia los cielos
English	La esperanza espera al otro lado, por eso volaré
English	Me pierdo en el camino cada vez que te busco
English	Siento los pensamientos que dejaste atrás
English	Me aferraré a ti y nunca te soltaré
English	Dos corazones que se buscan conforman este sueño

A líneas karaoke:

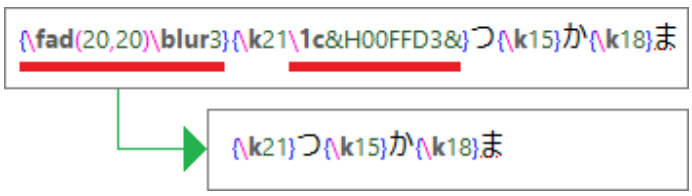
English	✱Mis ✱mejillas ✱se ✱manchan ✱con ✱lágrimas ✱de ✱soledad
English	✱Pero ✱puedo ✱sentir ✱la ✱llegada ✱del ✱amanecer
English	✱Que ✱me ✱atrae ✱con ✱rumbo ✱hacia ✱los ✱cielos
English	✱La ✱esperanza ✱espera ✱al ✱otro ✱lado, ✱por ✱eso ✱volaré
English	✱Me ✱pierdo ✱en ✱el ✱camino ✱cada ✱vez ✱que ✱te ✱busco
English	✱Siento ✱los ✱pensamientos ✱que ✱dejaste ✱atrás
English	✱Me ✱aferraré ✱a ✱ti ✱y ✱nunca ✱te ✱soltaré
English	✱Dos ✱corazones ✱que ✱se ✱buscan ✱conforman ✱este ✱sueño

```
{\k36}Me {\k145}aferraré {\k18}a {\k36}ti {\k18}y {\k91}nunca {\k36}te {\k127}soltaré
```

La función asigna un tiempo aproximado a cada palabra dependiendo de la cantidad de caracteres (letras) que tenga dicha palabra.

Y la otra función que cumple **text.kara( )** es remover todos los tags que no sean karaoke, de las líneas de karaoke:

Ejemplo:



text.tag(...)

Esta función agrega los tags que asignemos en sus parámetros, a cada uno de los caracteres (letras) del texto por default de un efecto.

Recordemos que el texto por default en un efecto depende del **Template Type**:

Template Type [fx]	Texto por Default
Syl	
Line	line.text_stripped
Word	word.text
Syl	syl.text
Furi	furi.text
Char	char.text
Convert to Hiragana	hira.text
Convert to Katakana	kata.text
Convert to Romaji	roma.text
Translation Line	line.text_stripped
Translation Word	word.text
Translation Char	char.text
Template Line [Word]	line.text_stripped
Template Line [Syl]	line.text_stripped
Template Line [Char]	line.text_stripped

Yo, por lo general uso esta función en los modos **Line** y **Translation Line**. Ya que esta función agrega tags a cada letra, entonces no tiene mucho sentido usarla en los modos **Char** y **Translation Char**.

Esta función no surge ningún efecto en los tres modos de **Template Line** (**[Word], [Syl] y [Char]**), y hay dos formas diferentes de agregar los tags a las letras:

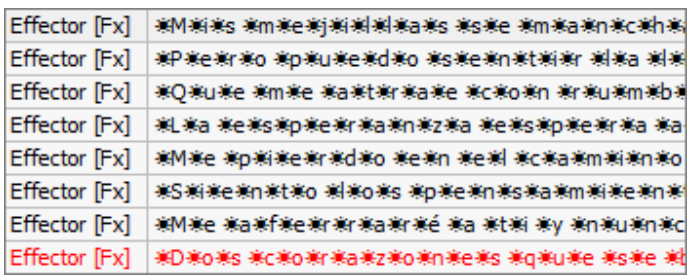
Ejemplo:

Template Type [fx]: Line

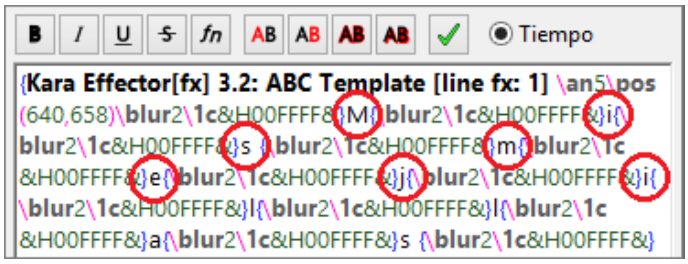
Return [fx]:

```
text.tag( "\\blur2", "\\1c&H00FFFF&" )
```

Como el ejemplo está hecho en modo **Line**, entonces en cada línea fx va la línea completa, y al aplicar el efecto con la función, entonces se le agregarán los tags ingresados a cada una de las letras de la línea de texto, como se nota en la siguiente imagen:



Acá vemos cómo los tags ingresados están antes de cada letra de la línea de texto:



La segunda forma de agregarle los tags a las letras, con esta función, es la más interesante:

Ejemplo:

Template Type [fx]: Line

Return [fx]:

```
text.tag( {"\\fscy", 120, 200},
          {"\\1c", "&H00FFFF&", "&H0000FF&"} )
```

Lo que hacemos es que, si un parámetro de la función es una tabla, debemos poner en el primer elemento de dicha tabla, al tag que queremos usar seguido de los dos valores a interpolar, el inicial y el final:

Vemos cómo el tag \fscy va creciendo desde 120 hasta 200, de letra a letra, y cómo el color primario \1c va cambiando de amarillo a rojo.

También se pueden combinar las dos formas de agregar los tags, y la cantidad de tags que se pueden añadir a la función es ilimitada.

Es todo por ahora para el **Tomo XXV**. Intenten poner en práctica todos los ejemplos vistos 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. 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