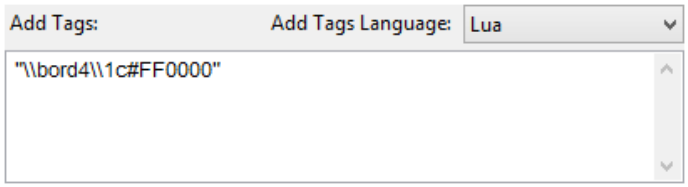
Kara Effector 3.2:

Para la fecha de publicación de este **Tomo VII** ya pueden descargar la más reciente actualización del **Kara Effector** que pone como fecha el 1 de Enero de 2014. La nueva actualización consiste en poder usar en nuestros Efectos los colores en formato **HTML** sin tener la necesidad de convertirlos a formato .ass y tener mayor diversidad de opciones al poder usar ambos formatos de manera libre.

La actualización fue inspirada al desarrollar el tema de las teorías de color que se usan en el **Aegisub** y les mostraré unos cuantos ejemplos de usarla:



Con esta actualización podemos copiar el código de un color en formato **HTML** y usarlo en el **Kara Effector**. En la web hay muchas páginas en donde podemos encontrar estos colores en dicho formato. Recomiendo la siguiente:

http://www.computerhope.com/htmcolor.htm

En esta web encontraremos un extenso listado de colores en formato **HTML** y así poder elegir el que más se ajuste al Efecto que aplicaremos:

#357EC7	Slate Blue
#488AC7	Silk Blue
#3090C7	Blue Ivy
#659EC7	Blue Koi
#87AFC7	Columbia Blue
#95B9C7	Baby Blue

Librería "random" [KE]:

Es una librería exclusiva del **Kara Effector** que consiste en una serie de funciones con resultados aleatorios que podemos aplicar en el desarrollo de los Efectos. La Librería "**random**" contiene las siguientes funciones:

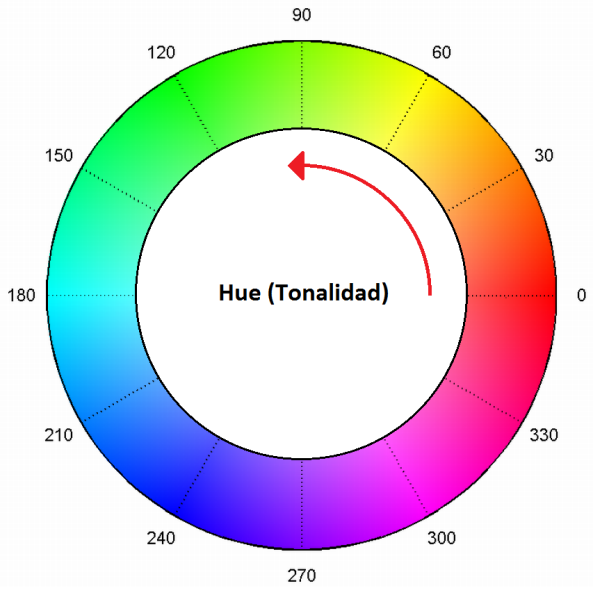
- **random.color**
- **random.colorvc**
- **random.alpha**
- **random.alphava**
- **random.e**
- **random.unique**

Y a continuación veremos en detalle en qué consiste casa una de ellas.

random.color(H, S, V): retorna un color en formato .ass según la teoría HSV. **H**, **S** y **V** son parámetros opcionales y pueden ser valores numéricos o tablas.

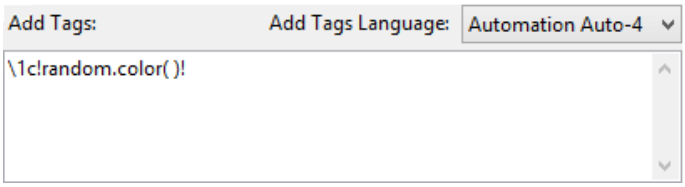
Modo 1: sin ningún parámetro

random.color(): retorna un color en formato .ass correspondiente a un tono (**Hue**) al azar entre 0 y 360 de la teoría **HSV**.



En este modo (Modo 1), la función retorna un tono al azar, pero tanto **S** (**Saturation**) como **V** (**Value**) son constantes y el valor de cada uno de ellos es 100, o sea que el color será solo alguno de los de la imagen anterior.

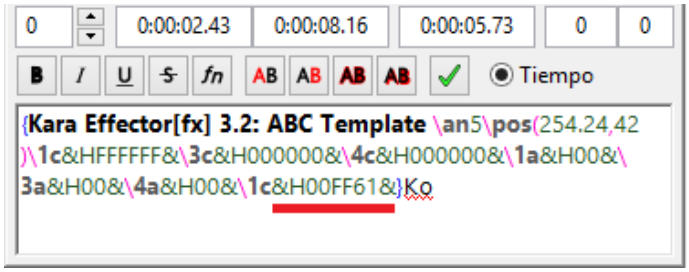
En el **Kara Effector**, en un Efecto tipo "**Syl**", podemos poner a prueba el Modo 1 de esta función:



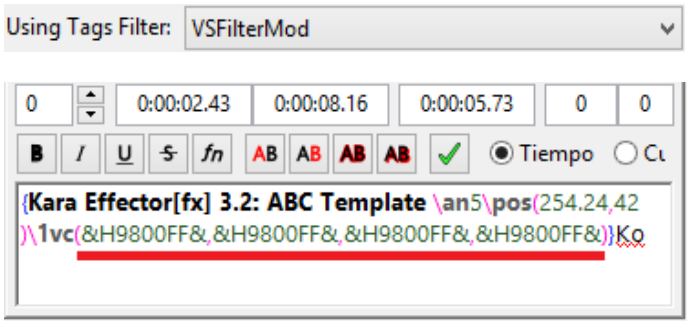
Entonces en el vídeo veremos algo como esto:

Para cada Sílaba la función generó un color aleatorio con un valor entre 0 y 360 del anterior círculo cromático.

Este es un ejemplo del color que puede retornar la función en este modo, siempre y cuando hayamos seleccionado la opción "**VSFilter 2.39**" o la opción "**No Tags Color and Alpha**" en **Using Tag Filter:**



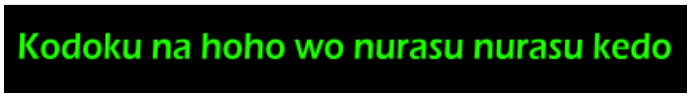
Si por el contrario, elegimos la opción "**VSFilterMod**", el color se multiplicará cuatro veces y saldrán separados por comas dentro de un paréntesis (en formato **VSFilterMod**):



Modo 2: con un parámetro constante

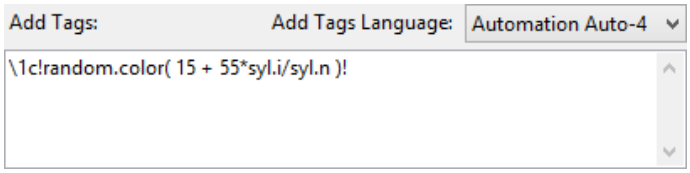
random.color(H): retorna un color en formato .ass correspondiente al tono (**Hue**) entre 0 y 360 que le hayamos asignado a la función. Ejemplo:

random.color(112) = "&H00FF21&"



En el círculo cromático vemos que el valor 112 le corresponde a un todo de verde claro. En este modo, la función retorna un color constante correspondiente al **Hue** del valor que le ingresemos.

Usar la función **random.color** para que retorne un color constante es ideal para hacer degradaciones con los colores **HSV**. Ejemplo: (**Template Type**: Syl)



Es un degradado **HSV** con los valores desde 15 para la primera Sílaba, hasta 70 (15 + 55 = 70) para la última:

Modo 3: con una Tabla como parámetro

random.color({H_i, H_f}): retorna un color en formato .ass correspondiente a un valor aleatorio entre **H_i** y **H_f**. Ejemplo:

random.color({270, 320}) = "&HFF00AA&"

Vemos que el tono (**Hue**) que resulta corresponde a un valor aleatorio entre 270 y 320 del círculo cromático **HSV**. O sea que cuando algún parámetro es una Tabla, entonces la función hace un Random entre los dos valores de la Tabla, cada vez que se ejecute la función.

Si combinamos los tres Modos de la función **random.color** (ausencia de uno o más de sus parámetros, uno o más parámetros constantes y por último, que uno o más de los parámetros sean una Tabla), obtenemos todo un mundo nuevo de posibilidades (15 en total):

- **random.color()**
- **random.color(H)**
- **random.color({H_i, H_f})**
- **random.color(H, S)**
- **random.color(H, {S_i, S_f})**
- **random.color({H_i, H_f}, S)**
- **random.color({H_i, H_f}, {S_i, S_f})**
- **random.color(H, S, V)**
- **random.color(H, S, {V_i, V_f})**
- **random.color(H, {S_i, S_f}, V)**
- **random.color({H_i, H_f}, S, V)**
- **random.color(H, {S_i, S_f}, {V_i, V_f})**
- **random.color({H_i, H_f}, S, {V_i, V_f})**
- **random.color({H_i, H_f}, {S_i, S_f}, V)**
- **random.color({H_i, H_f}, {S_i, S_f}, {V_i, V_f})**

En resumen:

- el parámetro **H** es un valor entre 0 y 360 o una tabla con dos elementos numéricos entre 0 y 360. Su valor por default es aleatorio entre 0 y 360.
- **S** y **V** son, o un valor entre 0 y 100 o una tabla con dos elementos numéricos entre 0 y 100. Sus valores por default son siempre de 100 para cada uno.

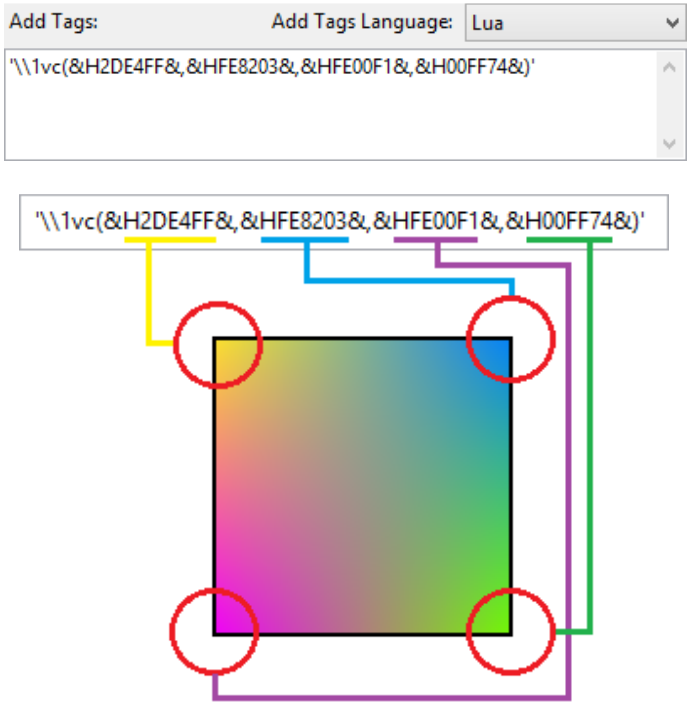
Ejemplos:

- **random.color(115)**
- **random.color({30, 220})**
- **random.color(304, 70)**
- **random.color(216, {60, 80})**
- **random.color({320, 340}, 90)**
- **random.color({125, 150}, {50, 100})**
- **random.color(0, 55, 92)**
- **random.color(86, 10, {0, 72})**
- **random.color(328, {60, 80}, 64)**
- **random.color({80, 120}, 77, 81)**
- **random.color(143, {0, 100}, {50, 80})**
- **random.color({45, 77}, 44, {66, 100})**
- **random.color({0, 105}, {70, 90}, 16)**
- **random.color({170, 204}, {55, 76}, {47, 88})**

Recomiendo poner a prueba en el **Kara Effector** cada uno de los anteriores ejemplos con el fin de practicar con esta función hasta que se familiaricen con cada uno de sus modos y opciones de uso. Esta es la primera función de la Librería "**random**" y restan cinco más por ver.

random.colorvc(H, S, V): es similar a **random.color** con la única diferencia que retorna un color en formato del filtro **VSFiltermod**.

Un color en formato **VSFilterMod** está compuesto por cuatro colores para cada uno de sus cuadrantes. Ejemplo:



O sea que todo color en formato **VSFilterMod** tiene la siguiente estructura:

(color SI, color SD, color II, color ID)

- **color SI:** Superior Izquierdo
- **color SD:** Superior Derecho
- **color II:** Inferior Izquierdo
- **color ID:** Inferior Derecho

Y lo que hace la función **random.colorvc** es usar la función **random.color** para generar a cada uno de esos cuatro anteriores colores. Los modos y todas las combinaciones posibles de la función **random.colorvc** son exactamente los mismos que en **random.color**, vista anteriormente:

- **random.colorvc()**
- **random.colorvc(H)**
- **random.colorvc({H_i, H_f})**
- **random.colorvc(H, S)**
- **random.colorvc(H, {S_i, S_f})**
- **random.colorvc({H_i, H_f}, S)**
- **random.colorvc({H_i, H_f}, {S_i, S_f})**
- **random.colorvc(H, S, V)**
- **random.colorvc(H, S, {V_i, V_f})**
- **random.colorvc(H, {S_i, S_f}, V)**
- **random.colorvc({H_i, H_f}, S, V)**
- **random.colorvc(H, {S_i, S_f}, {V_i, V_f})**
- **random.colorvc({H_i, H_f}, S, {V_i, V_f})**
- **random.colorvc({H_i, H_f}, {S_i, S_f}, V)**
- **random.colorvc({H_i, H_f}, {S_i, S_f}, {V_i, V_f})**

No sobra recordarles que para usar colores en formato **VSFilterMod** debemos seleccionar esa opción en **Using Tags Filter**, de otro modo el **Kara Effector** convertirá el color a formato **VSFilter 2.39**.

---

random.alpha(A1, A2): retorna una transparencia (**alpha**) aleatoria en formato .ass (**VSFilter 2.39**) desde el valor de **A1** hasta **A2**.

Los parámetros **A1** y **A2** pueden ser números reales entre 0 y 255, o valores **alpha** entre "&H00&" y "&HFF&".

Modo 1: sin parámetros

random.alpha(): retorna una transparencia al azar entre totalmente visible (0 en decimal o "&H00" en formato .ass), hasta totalmente invisible (255 en decimal o "&HFF&" en formato .ass).

Modo 2: parámetros numéricos

random.alpha(A1, A2): retorna una transparencia aleatoria entre los valores numéricos de **A1** y **A2**. Ejemplos:

- **random.alpha(55, 142)**
- **random.alpha(0, 200)**
- **random.alpha(180, 255)**

Modo 3: parámetros **alpha** en formato .ass

random.alpha(A1, A2): retorna una transparencia aleatoria entre los valores **alpha** (.ass) de **A1** y **A2**. Ejemplos:

- **random.alpha("&HA3&", "&HED&")**
- **random.alpha("&H0C&", "&H7F&")**

Una especie de Modo 4 sería combinar los modos 2 y 3. Ejemplos:

- **random.alpha(120, “&HED&”)**
- **random.alpha(“&H0C&”, 215)**

---

random.alphava(A1, A2): es muy similar a la anterior función y retorna una transparencia (**alpha**) aleatoria en formato **VSFilterMod** desde el valor de **A1** hasta **A2**. Los Modos y formas de uso son los mismos que usamos en **random.alpha**

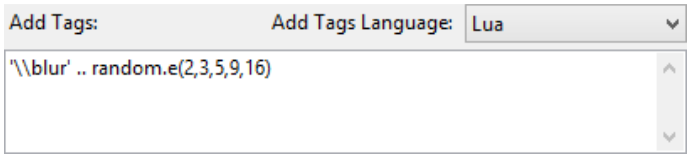
Para cada una de las cuatro transparencias (alpha), que conforman a una transparencia en formato **VSFilterMod**, se ejecuta el random que hace que sea prácticamente imposible que las cuatro sean iguales. Ejemplos:

- **random.alphava(90, 180)**
- **random.alphava(0, 100)**
- **random.alphava(“&HC4&”, “&HFF&”)**
- **random.alphava(“&H00&”, “&H86&”)**
- **random.alphava(12, “&HAA&”)**
- **random.alphava(“&HDF&”, 125)**

---

random.e(…): retorna un parámetro al azar de entre todos los que se le hayan ingresado o un elemento al azar, entre los elementos de una tabla que se le haya ingresado como parámetro.

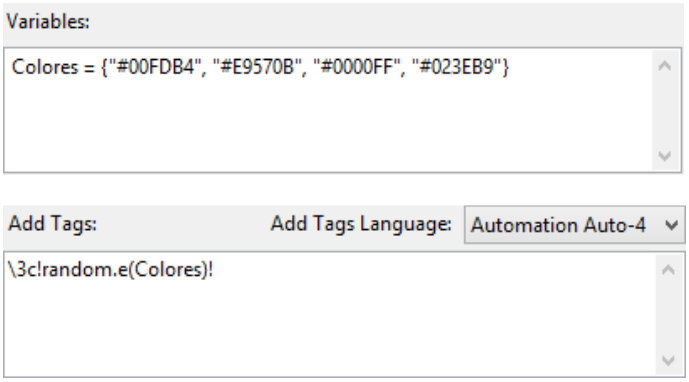
Ejemplo 1:



La función retornaría alguno de esos cinco valores que se le ingresaron, seleccionado de forma aleatoria, o sea que los posibles resultados serán:

- \blur2
- \blur3
- \blur5
- \blur9
- \blur16

Ejemplo 2:



Declaré una tabla llamada “Colores”, entonces la función **random.e** selecciona un elemento al azar de entre los cuatro que tiene la tabla.

---

random.unique(T, i): primero desordena la posición de los elementos de la tabla **T** para posteriormente retornar al elemento **T[i]**. **T** puede ser una tabla propiamente dicha o un número entero positivo. En el caso de que **T** sea un entero, entonces se crea una tabla con los números desde 1 hasta **T**.

Ejemplo 1:

random.unique(syl.n, syl.i): primero crea una tabla con los números consecutivos desde 1 hasta syl.n:

T = {1, 2, 3, 4, 5, 6, 7, 8, … , syl.n}

Luego desordena la posición de los elementos de la tabla de manera aleatoria:

T = {5, 8, 4, syl.n, 2, 6, 3, … , 1, 7 }

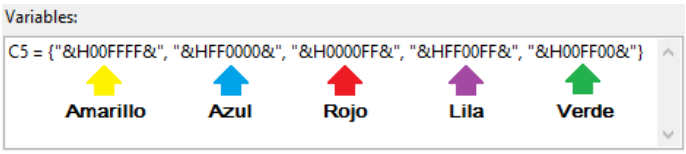
Por último, retorna **T**[syl.i]:

- **T**[1] = 5
- **T**[2] = 8
- **T**[3] = 4
- **T**[4] = syl.n
- **T**[syl.n - 1] = 1
- **T**[syl.n] = 7

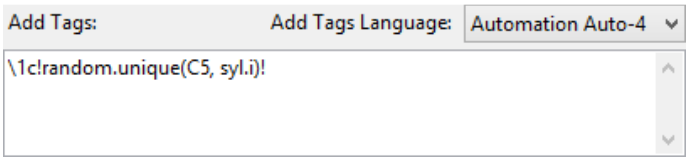
O sea que la función **random.unique** nunca repite un resultado, a menos que sea usada una cantidad mayor de veces que el tamaño de la tabla.

Ejemplo 2:

Declaramos una tabla con cinco colores dispuestos de la siguiente forma:



Si intentamos usar un color para cada Sílaba, y si una Línea de Texto tiene más de cinco Sílabas, entonces esto hace imposible que la función **random.unique** le asigne un único color de esta tabla a cada una de las Sílabas. Lo que hace la función es agotar las opciones y luego repite el mismo patrón una y otra vez:



Como la tabla "**C5**" de nuestro ejemplo solo tiene cinco elementos, el resultado **C5**[6] o cualquier otro mayor no existiría, entonces la función repite los resultados:

Para esta línea de texto la función reorganizó los colores de la tabla en forma aleatoria así:

1. Rojo
2. Amarillo
3. Lila
4. Verde
5. Azul

Y vemos que el patrón, al menos en esta línea, se repitió tres veces:

Esta se podría considerar como la función más compleja de la Librería "**random**", pero con un poco de práctica se irán acostumbrando y encontrarán la forma de darle una aplicación en algunos de sus Efectos y proyectos.

Con la conclusión de la Librería "**random**" se da por terminado el **Tomo VII**, con la recomendación de poner en práctica los ejemplos vistos en él. No olviden descargar la más reciente 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.