Actualización R(Val1, Val2, Step)

El **Kara Effector** ha venido evolucionando de forma exponencial, y por eso la necesidad de tener un material de apoyo que nos guie y ayudé a sacarle el mayor provecho a dichas evoluciones.

En este artículo veremos las funciones que se han actualizado, en qué consisten y cómo se aplican, y ejemplo de la forma en la que podemos usarla; también veremos la evolución que han tenido las celdas de texto de la **Ventana de Modificación** del **KE**, para que sea aún más simple el poder agregar funciones, tags, entre otras herramientas disponibles para nuestros fx.

Recursos [KE]:

La primera función que me interesa que vean la forma en la que ha evolucionado es la función **R**
(es la versión **math.random**
del **KE**), que no es otra que aquella que nos permite obtener un número aleatorio o al azar, siguiendo una serie de parámetros o argumentos predeterminados, o por default si así lo queremos.

Actualización: R(Val1, Val2, Step)

Recordemos que esta función retorna un número aleatorio que se encuentre entre los argumentos **val1** y **val2**, inclusive, en un paso determinado por el tercer argumento, **Step**.

Ejemplo:

R(0, 100, 10)

De este modo, la función retornará un número entero entre 0 y 100 de a 10 en 10, es decir que los posibles resultados serían:

0, 10, 20, 30, 40, 50, 60, 70, 80, 90 o **100**

Partiendo de este hecho, y de que la función **R**
es una de las más usadas a la hora de desarrollar un efecto, hemos ampliado su rango de operación, para así ampliar nuestras posibilidades, agregando una serie de letras para que la función cumpla con otros requisitos.

Las actualizaciones son 16 en total, y veremos ejemplos de cada una de ellas para comprender en qué consisten:

- **Rs(Val1, Val2, Step)**: la letra "**s**" es de "**signo**", es decir que esta función retorna un número aleatorio entre **Val1** y **Val2**, y el signo de dicho número también será asignado de forma aleatoria. Recordemos que los tres argumentos de esta función son opcionales, de manera que si solo ponemos un solo argumento, el random original se generará entre el 1 y dicho argumento y el valor del **Step** será 1 por default. Si ponemos dos argumentos, el random se llevará a cabo entre esos dos valores inclusive, y el **Step** también será 1.

Ejemplo:

Rs(12, 18)

Retornará un número aleatorio entre 12 y 18 con cualquiera de los dos signos, positivo o negativo. Entonces los posibles resultados de este ejemplo serían:

–**18, -17, -16, -15, -14, -13, -12**

12, 13, 14, 15, 16, 17 o **18**

- **Rr(Val1, Val2, Step)**: la letra "**r**" es de "**ratio**". La función retornará un número aleatorio que esté entre los primeros dos argumentos inclusive, pero el resultado final dependerá del tamaño del vídeo en dónde se aplique el efecto, es decir que si el vídeo es de 720p el resultado no se modificará, pero si es más pequeño, el resultado disminuirá proporcionalmente; y pasaría lo mismo si el vídeo es de mayor resolución que el ya mencionado.

Ejemplo:

Rr(8, 10)

Los tres posibles resultados de este ejemplo en un vídeo de 720 x 1280 px son 8, 9 o 10, pero si el vídeo fuera de 1080 x 1960 px, cualquiera de esos tres resultados se multiplica por el "ratio", es decir, la razón entre 1080 y 720:

1080 / 720 = 1.5

O sea que los tres posibles resultados para el vídeo de 1080p vendrían a ser:

- 8 * 1.5 = 12
- 9 * 1.5 = 13.5
- 10 * 1.5 = 15

Esta actualización tiene su aplicación, por ejemplo, al designar aleatoriamente el borde con el tag \bord; ya que un \bord3 para un vídeo de 720p puede que sea muy poco para un vídeo de 1080p, o muy poco para uno de 480p.

- **Rd(Val1, Val2, Step)**: la letra "**d**" es de "**décima**". La función retorna un valor entre los dos primeros argumentos, pero redondeados al primer decimal.

Ejemplo

Rd(-1, 2)

La función retornaría un número entre -1 y 2, pero tendría en cuenta los números redondeados al primer decimal, o sea que los resultados serían:

-1, -0.9, -0.8, -0.7, -0.6, -0.5, -0.4, -0.3, -0.2, -0.1, 0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1, 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.8 o **2**

  - **Rc(Val1, Val2, Step)**: la letra "**c**" es de "**centésima**". Es similar a la actualización anterior, pero con la diferencia que el valor retornado estará redondeado al segundo decimal, es decir que la cantidad de opciones posibles se amplían 100 veces más que en la función normal.

- **Rm(Val1, Val2, Step)**: la letra "**m**" es de "**milésima**". Redondea el número a la tercera cifra decimal.

Para las siguientes actualizaciones, lo único que debemos hacer es combinar las letras aprendidas en las actualizaciones anteriores de la función **R**, de la siguiente forma:

- **Rsr** o **Rrs**: Random + signo + ratio
- **Rdr**: Random + décima + ratio
- **Rcr**: Random + centésima + ratio
- **Rmr**: Random + milésima + ratio
- **Rds**: Random + décima + signo
- **Rcs**: Random + centésima + signo
- **Rms**: Random + milésima + signo
- **Rdrs**: Random + décima + ratio + signo
- **Rcrs**: Random + centésima + ratio + signo
- **Rmrs**: Random + milésima + ratio + signo

Y la última de las actualizaciones de la función **R**
es una que nos permite ingresarle una **tabla** y obtendremos uno de sus elementos seleccionados de forma aleatoria:

- **Re(tabla)**: la letra "**e**" es de "**elemento**". Retorna un elemento seleccionado aleatoriamente de la **tabla** que le ingresemos como su único argumento. Tiene la ventaja que también lo puede hacer en **tablas** que no sean indexadas:

Ejemplo:

```
my_table = {

   [1] = 4,

   [2] = "\\fry-45",

   Size = Rr( 80, 100 ),

   80,

   -12

}
```

después de indicar los elementos de la tabla se puede utilizar de este modo:

Re(my_table)

Así, de este modo, la función podría retornar cualquiera de los cinco elementos de la **tabla** "**my_table**", es decir: **4, \\fry-45, Rr(80, 100), 80** o **-12**

Con todas estas modificaciones, creo que quedan cubiertas muchas o gran parte de todas las posibilidades a la hora de generar un valor aleatorio. Lo que viene de acá en más es poner en práctica y experimentar con lo aprendido e ir desarrollando nuestros propios efectos y estilos.

El tema que veremos a continuación, considero que es uno de los más importantes para todos aquellos que les gusta desarrollar y llevar a cabo aquellos efectos que se les vienen en mente o que se presentan en los diferentes vídeos y que desean de alguna manera poder “imitarlos” para sus propios proyectos. El tema consiste es una serie de abreviaciones, convenciones y un par de trucos más que el **KE** tiene en sus librerías para que lo que antes eran simples tags, ahora se conviertan en funcionen, y por qué no, en efectos en sí mismos.

Para continuar con la lectura debes de ir al articulo Herramienta: tags abreviados y tag funciones.

Muchas gracias por seguirnos.