

Creamos un tema con SuperKaramba

WIDGETS CREATIVOS



Si no encontramos el tema de SuperKaramba que estábamos buscando, siempre podremos fabricarnos el nuestro propio.

POR HAGEN HÖPFNER

Un bonito fondo de pantalla está bien hasta cierto punto, pero si queremos adornos de escritorio que realmente nos ayuden con nuestro trabajo, como mostrar previsiones meteorológicas o monitorizar nuestro sistema, necesitaremos una herramienta como SuperKaramba. SuperKaramba [1] es una herramienta basada en KDE que nos permite crear útiles widgets para nuestro escritorio. Su página se refiere a estos widgets interactivos como “caramelos para los ojos”, aunque sus applets se denominan comúnmente *temas*. Podemos encontrar muchos temas de SuperKaramba ya elaborados en Internet, pero también podemos crear los nuestros. Un tema puede tomar forma de juego, monitor de sistema, directorio de música del escritorio o incluso una barra de herramientas personalizada.

En anteriores artículos de Linux Magazine hemos descrito cómo iniciarse con SuperKaramba. Para información adicional, véase [2]. (Estos artículos están disponibles en el magnífico archivo de artículos de Linux Magazine). El presente artículo es un breve taller acerca de cómo crear nuestro propio tema para SuperKaramba.

Encontrar el Software

La versión actual de SuperKaramba es la 0.37. Si preferimos no compilarlo no-

sotros mismos, encontraremos archivos RPM y Debin en [3] y [4]. Al ejecutar *superkaramba* se abre un cuadro de diálogo para los temas (véase Figura 1) y se despliega un icono de control dentro de la barra de KDE.

SuperKaramba necesita los temas para llenar de vida el escritorio. Los diseños están disponibles para muchas aplicaciones típicas. Al presionar el botón *New Themes...* se abre un cuadro de selección que nos ofrece un buen número de temas listados en [5]. Si seleccionamos una entrada, podemos hacernos una primera idea de la información acerca de la funcionalidad y una captura de pantalla. SuperKaramba descarga los archivos necesarios y los instala en el directorio `~/.kde/share/apps/superkaramba/themes`. Al cerrar el cuadro de diálogo de los temas, los instalados aparecen en la ventana de SuperKaramba, desde donde podremos lanzarlos.

Podemos ejecutar varios temas en paralelo en el mismo escritorio, ejecutando o eliminando cada tema según se requiera. El menú del botón derecho del ratón también nos permite especificar si queremos mantener la posibilidad de moverlo. Algunos temas tienen sus propios cuadros de diálogo para la configuración. Por ejemplo, podemos configurar un parte meteorológico para que se ajuste a nuestra ubicación.

Crear Nuestros Propios Temas

Para crear nuestro propio tema, necesitamos crear un archivo, supongamos *mi_tema.theme*, el cual podemos abrir desde SuperKaramba como un archivo local. Como se pueden añadir gráficos y scripts Python en el archivo de diseño, probablemente vamos a querer guardar todos estos componentes en un único directorio.

El archivo *theme* comprende tres componentes: comandos generales, que especifican la geometría y definen las zonas interactivas. Los sensores ayudan a leer parámetros del sistema, como la carga actual de la CPU. Por último, los medidores muestran los valores leídos. El Listado 1 ilustra el aspecto de un archivo *theme*.

La primera línea del Listado 1 sitúa el tema en la esquina inferior izquierda del escritorio, fija un ancho de 200 píxeles y una altura de 400 píxeles, especifica que el tema puede moverse y fija un tiempo de refresco de 1000 milisegundos. Podríamos configurar la posición del tema especificando un offset *x* e *y* medidos desde la esquina superior izquierda.

El parámetro *BOTTOM = true* ignora la posición horizontal. De manera similar, la opción *RIGHT = true* sitúa el tema en el

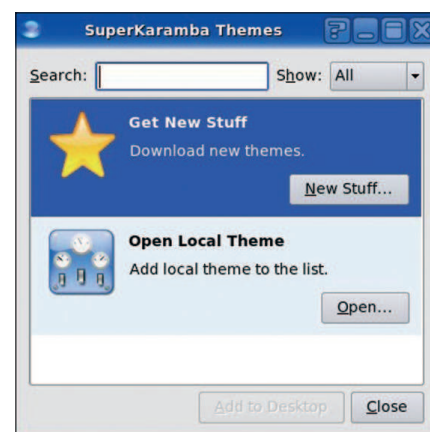


Figura 1: Usamos el cuadro de diálogo de temas de SuperKaramba para instalar temas nuevos.



Figura 2: Antes de comenzar a crear nuestros temas SuperKaramba, puede ser buena idea echarle un vistazo a los temas disponibles en Internet.

margen derecho. `ONTOP = true` evita que otras ventanas oculten el tema. `TOPBAR = true` y `BOTTOMBAR = true` anclan el tema en la parte superior o inferior de la pantalla, de manera similar a la barra de KDE, sin que las ventanas maximizadas puedan ocultarlo.

La segunda línea del Listado 1 fija un tipo de letra por defecto, un tamaño y color para todos los elementos de texto asociados con el tema. El código de color define una combinación de valores para el rojo, verde y azul, con un valor máximo de 255. En el ejemplo del Listado 1, se usa `255,255,255`, que es el código que representa el blanco.

Sensores y Medidores

Las líneas 5 y 6 del Listado 1 crean dos medidores de texto que derivan su contenido del sensor de la hora (`sensor = time`). La línea 5 genera el hora



Figura 3: Gracias a la interfaz Phyton, programar temas con funcionalidades externas es rápido y sencillo.

El primer grupo de nuestro tema de ejemplo está situado en $x = 10$ e $y = 10$. Además del medidor de la hora y de la fecha, contiene una definición del área interactiva. La línea 7 le indica a SuperKaramba que llame al comando de configuración de la hora en KDE cuando se haga doble clic en el área de 120 por 34 píxeles. El parámetro `preview = true` dibuja un marco alrededor del tema para indicar que aún está de prueba.

El parámetro `format` define el contenido generado y el formato de un campo en función del sensor que hayamos soli-

y hace caso omiso del tamaño de letra por defecto, que se definió con anterioridad.

Es una buena idea usar las etiquetas `<GROUP>` para agrupar elementos similares. Esto mejora la legibilidad y nos permite especificar una posición para el grupo.

El primer grupo de nuestro

tema de ejemplo está situado en $x = 10$ e $y = 10$. Además del medidor de la hora y de la fecha, contiene una definición del área interactiva. La línea 7 le indica a SuperKaramba que llame al comando de configuración de la hora en KDE cuando se haga doble clic en el área de 120 por 34 píxeles. El parámetro `preview = true` dibuja un marco alrededor del tema para indicar que aún está de prueba.

- `cpu`: Carga del sistema
- `disk`: Uso del disco para sistemas de archivo montados
- `memory`: Memoria libre y usada
- `network`: Tráfico de red de entrada y salida
- `noatun`: Información de un proceso noatun en ejecución
- `program`: Salida estándar para cualquier programa
- `sensor`: Procesa la salida del sensor LM [10]
- `textfile`: Lee un archivo de texto de manera continua
- `uptime`: tiempo en servicio del sistema
- `xmms`: Información de un proceso XMMS

El segundo grupo del Listado 1 ilustra el uso de alguno de los sensores en combinación con varios dispositivos de salida. La línea 12 lee la memoria libre en Megabytes sin contar el búfer y la memoria caché. La línea 14 usa la imagen `bar.png` para crear un diagrama de barras que indique la carga actual de la CPU.

Tabla 1: Funciones Callback de SuperKaramba

Función	Evento	Parámetros pasados
<code>initWidget(widget)</code>	Crear el widget de SuperKaramba	
<code>widgetUpdated(widget)</code>	Actualizar el tema	Intervalo de actualización del archivo <code>.theme</code>
<code>widgetClicked(widget, x, y, button)</code>	Clic del ratón en el tema	<code>x</code> e <code>y</code> : coordenadas de un clic relativo al tema; <code>button</code> : botón del ratón usado
<code>widgetMouseMoved(widget, x, y, button)</code>	Movimiento del ratón dentro de un tema	<code>x</code> e <code>y</code> : coordenadas de un clic relativo al tema; <code>button</code> : botón del ratón usado
<code>menuItemClicked(widget, menu, id)</code>	Clic en el ítem del menú	<code>menu</code> : controlador del menú; <code>id</code> : controlador del ítem del menú
<code>menuItemOptionChanged(widget, key, value)</code>	Ítem del menú de configuración llamado en el tema	<code>key</code> : controlador del ítem del menú; <code>value</code> : nuevo valor para el ítem del menú (verdadero o falso)
<code>meterClicked(widget, meter, button)</code>	Clic en un medidor	<code>meter</code> : controlador del dispositivo medidor; <code>button</code> : botón del ratón usado
<code>itemDropped(widget, dropText)</code>	Objetos desplegados en el tema en operaciones de arrastrar y soltar	<code>dropText</code> : texto del objeto (vg: URL)
<code>startupAdded(widget, startup)</code>	KDE lanza una aplicación	Tras completarse la inicialización, le siguen las señales <code>startupRemoved()</code> y <code>taskAdded()</code>
<code>startupRemoved(widget, startup)</code>	Véase <code>startupAdded()</code>	
<code>taskAdded(widget, task)</code>	Véase <code>startupAdded()</code>	
<code>taskRemoved(widget, task)</code>	Fin del programa de aplicación	
<code>activeTaskChanged(widget, task)</code>	Mueve una aplicación a primer plano	

Listado 1: mi_tema.theme

```

01 KARAMBA x=0 BOTTOM=true w=200 h=400 LOCKED=false INTERVAL=1000
02 DEFAULT font="Sans" fontsize=10 shadow=2 color=255,255,255
03
04 <GROUP> x=10 y=10
05     TEXT x=12 y=0 sensor=time fontsize=12 format="hh:mm:ss"
06     TEXT x=12 y=15 sensor=time format="ddd dd.MM.yyyy"
07 CLICKAREA x=0 y=0 w=120 h=34 onclick="kdesu kcmsHELL clock"
08 </GROUP>
09
10 <GROUP> x=10 y=50
11     TEXT x=12 y=0 value="MEM"
12     TEXT x=45 y=0 sensor=memory format="%fmb MB"
13     TEXT x=12 y=15 value="CPU"
14     BAR x=45 y=15 sensor=cpu path="bar.png"
15     TEXT x=12 y=30 value="IN"
16     GRAPH x=45 y=30 h=12 w=70 color=255,255,255 points=100
        sensor=network device="eth0" format="%in"
17     IMAGE x=0 y=50 path="background.png"
18 </GROUP>

```

Para evaluar el tráfico entrante en la línea 16, se usa un gráfico de 12 x 70 píxeles de color blanco. La última línea en este grupo usa el dispositivo de salida *IMAGE* para mostrar una imagen. Opcionalmente, *PATH* acepta una URL, que nos permite incrustar una imagen de Internet. La documentación de las opciones de formato para los distintos dispositivos de salida está disponible en [9].

SuperKaramba y Python

La interfaz Python [11] nos ayuda a conseguir un tema mucho más flexible. Las funciones de callback añaden la posibilidad de reaccionar ante eventos disparados por el escritorio KDE (véase la Tabla 1). Para

permitir que un tema acceda a un archivo Python, éste debe estar ubicado en el mismo directorio que el archivo *theme*. Debe también usar el mismo nombre pero con la extensión *.py*. La plantilla en [12] es un útil punto de partida para nuestro propio desarrollo, ya que integra un buen número de funciones callback.

El Listado 2 muestra un script Python que permite al usuario reemplazar la imagen mostrada en el tema SuperKaramba mediante arrastrar y soltar. Comienza importando módulos adicionales: *import karamba* carga el módulo de SuperKaramba e *import string* añade las funciones de manipulaciones de cadenas para las variables de texto del script.

Listado 2: mi_tema.py

```

01 import karamba # Imports the Karamba API
02 import string # String manipulation functions
03 image=0
04
05 def initWidget(widget):
06     global image
07     karamba.acceptDrops(widget)
08     image=karamba.createImage(widget, 12, 102, "bar")
09     karamba.hideImage(widget, image)
10     karamba.redrawWidget(widget)
11
12 def itemDropped(widget, dropText):
13     global image
14     image_link=string.split(dropText, "file:", 1)[1]
15     karamba.deleteImage(widget, image)
16     image=karamba.createImage(widget, 12, 102, image_link)
17     karamba.resizeImage(widget, image, 176, 286)

```

La función callback *initWidget(widget)* se llama automáticamente cuando se genera la ventana del tema. *karamba.acceptDrops(widget)* configura el widget para que acepte imágenes que el usuario puede arrastrar y soltar en la ventana. Al mismo tiempo, *karamba.createImage* crea una imagen que se usa como parámetro de sustitución, lo cual explica porqué está oculta al principio (*karamba.hideImage(widget, image)*) hasta que *karamba.redrawWidget(widget)* refresca la pantalla.

La segunda función callback en el Listado 2 es *itemDropped(widget, dropText)*, que se llama cuando el usuario suelta un objeto en el widget. La variable *dropText* proporciona la URL para el objeto. La siguiente línea destripa el archivo: le añade un prefijo y descubre la ruta hasta el archivo local. Entonces la función elimina la imagen original de la ventana del tema y muestra la nueva imagen, usando la última línea para escalar la imagen (véase la Figura 3). ■

Recursos

- [1] SuperKaramba: <http://netdragon.sourceforge.net/ssuperkaramba.html>
- [2] "KTools: Decoración de Escritorios", por Stefanie Teufel, Linux Magazine Edición en Castellano, Número 4, p. 67. <http://www.linux-magazine.es/issue/04/SuperKaramba.pdf>
- [3] Página del guru en RPM: <http://linux01.gwdg.de/~pbleser>
- [4] Paquetes SuperKaramba para Debian: <http://archive.linux-peter.de/debian/pool/main/s/superkaramba>
- [5] Temas SuperKaramba: <http://www.kde-look.org>
- [6] COMO acerca de crear un tema: <http://netdragon.sourceforge.net/screate.html>
- [7] Crear temas no rectangulares: <http://netdragon.sourceforge.net/smasks.html>
- [8] Resumen general de sensores: <http://netdragon.sourceforge.net/ssensors.html>
- [9] Soporte para Medidores: <http://netdragon.sourceforge.net/smeters.html>
- [10] Sensores LM: <http://secure.netroedge.com/~lm78>
- [11] SuperKaramba Python API: <http://netdragon.sourceforge.net/api.html>
- [12] Plantilla Python para SuperKaramba: <http://netdragon.sourceforge.net/template.py>