2.7. LA CLASE EventType Y EL MÉTODO bind

Algunos de los widgets estudiados reconocen la opción command con la que se especifica que acción realiza el widget en respuesta a eventos del usuario, como pulsar un botón con el ratón o con la barra espaciadora si tiene el foco del teclado, o seleccionar una opción de una lista con los mismos eventos.

Otros widgets no tienen esta opción porque, por defecto, no se espera que el usuario interactúe con ellos. Pero es posible que en un caso específico queramos que una etiqueta que muestra un enlace web responda a un clic del usuario lanzando el navegador con la página web enlazada. Para conseguirlo basta con conectar el evento "clic sobre el widget" con una función que realice la acción deseada.

El módulo webbrowser se encarga de abrir el navegador por defecto y mostrar la web en una nueva pestaña.

```
from tkinter import Tk, Label
import webbrowser as wb
def f(e):
    wb.open(l['text'], 2)
v = Tk()
l = Label(v, text='http://umh.es', fg='blue', cursor='hand')
l.bind('<Button-1>', f)
l.pack()
v.mainloop()
```

La función bind está declarada en la clase Misc por lo que la pueden usar todos los widgets. El primer parámetro es una cadena de caracteres que identifica al evento, como `widget.bind(evento, acción)` '<Button-1>' que significa que "se ha hecho clic con el botón principal del ratón". El segundo parámetro es el nombre de la función que se ha de ejecutar cuando el widget reciba el evento, función que ha de tener un parámetro que recoge información sobre el evento.

Los eventos se identifican mediante una cadena de caracteres con el patrón <MODIFICADOR-MODIFICADOR-TIPO-DETALLE>, donde MODIFICADOR puede ser

Control Mod2 M2 Shift Mod3 M3 Lock Mod4 M4
Button1 B1 Mod5 M5 Button2 B2 Meta M Button3
B3 Alt Button4 B4 Double Button5 B5 Triple Mod1
M1

y sirve para añadir hasta dos modificaciones, Alt+Shift p.ej., al TIPO

Activate Enter Map ButtonPress Button Expose Motion
ButtonRelease FocusIn MouseWheel Circulate FocusOut
Property Colormap Gravity Reparent Configure KeyPress
Key Unmap Deactivate KeyRelease Visibility Destroy Leave

pudiendo ser el DETALLE el número del botón para los eventos ButtonPress y ButtonRelease o el símbolo de tecla para KeyPress y KeyRelease.

Por ejemplo, <Button-1> es el evento que recibe un widget si hacemos clic sobre él con el botón izquierdo (por defecto) del ratón. Es el mismo evento que <ButtonPress-1>, <Button-1> o <1>. Cuando el usuario pulsa el botón, tkinter sabe que es posible que el usuario siga haciendo algo con el ratón, p.ej. moverlo o soltar el botón, por lo que se guarda la posición donde se hizo clic por si tiene que usarla. Con <2> y <3> (o sinónimos) tendremos el mismo control sobre el botón central y derecho que suelen tener los ratones. Con <B1-Motion>, si el usuario está pulsando el botón izquierdo podemos capturar la posición del cursor mientras se mueve. <ButtonRelease-1> nos informa de la posición en que está el cursor cuando se deja de apretar el botón izquierdo.

La función asociada al evento recibe un parámetro con información del evento. Para averiguar qué información recibe la función podemos pedirle que nos muestre su diccionario. Observaremos que hay información sobre el teclado (keycode, char, keysym), sobre la posición del cursor del ratón (x, y), el widget que recibe el evento, el tipo de evento... Tomando lo que necesitemos de esta información podemos, por ejemplo, mostrar en el título de la ventana la posición del cursor al hacer clic sobre ella y mover el ratón sin soltar el botón:

```
def f(e):
    print(str(e.__dict__))
```

```
from tkinter import Tk
v = Tk()
v.bind('<B1-Motion>', lambda e: v.title('(' + str(e.x) + str(e.y) + ')'))
v.mainloop()
```

Los eventos <Enter> y <Leave> se lanzan cuando el cursor "entra" y "sale" del widget. <FocusIn> y <FocusOut> se lanzan cuando el widget recibe o pierde el foco del teclado. <Configure> se lanza cuando se modifica la configuración del widget, p.ej. al redimensionarlo.

<Key> se lanza al pulsar cualquier tecla, pudiendo capturar cuándo se pulsa y cuándo se suelta con <KeyPress> y <KeyRelease>. Si queremos detectar una tecla concreta no necesitamos rodearla de < y >, basta con escribir el carácter que representa, p.ej. a, siendo 1 el evento "se ha pulsado la tecla 1" y <1> el evento "se ha pulsado el botón izquierdo del ratón". Las ausencias de teclas tiene un evento que la identifica, como <space>, <less>, <Return>, <Cancel>, <BackSpace>, <Tab>, <Shift_L>, <Control_L>, <Alt_L>, <Pause>, <Caps_Lock>, <Escape>, <Prior>, <Next>, <End>, <Home>, <Left>, <Up>, <Right>, <Down>, <Print>, <Insert>, <Delete>, <F1> ... <F12>, <Num_Lock> y <Scroll_Lock>. Puedes averiguar qué símbolo tiene asociado cada tecla añadiendo estas líneas al código anterior: `v.bind('<Key>', lambda e: print(e.keysym))`

Si una vez capturado el evento no queremos que se propague y haga lo que tkinter ya tiene codificado, la función asociada ha de devolver el string 'break'.