

Programando con Mozilla Xul

Clics Bonitos

En los primeros capítulos de nuestra serie de XUL hemos estado principalmente mirando los conceptos básicos del lenguaje. A medida que nuestro viaje ha continuado hemos explorado botones, etiquetas y gestión de geometrías, comenzando el mes pasado conectar funcionalidad a nuestro interfaz.

POR JONO BACON



Este mes vamos a repasar con mucha atención qué menús se utilizan y para qué y como crearlos. Una vez hecho esto juntaremos toda la información de los últimos números para crear un interfaz XUL completo. Si bien este interfaz no incluirá toda la funcionalidad de una aplicación completa, sí nos permitirá asentar nuestros conocimientos de XUL mediante la creación de un interfaz completo y práctico.

Los menús proporcionan una forma de ocultar funcionalidad manteniéndola accesible de forma sencilla para el usuario. El modo de funcionar de los menús es muy ingeniosa. Si necesitásemos mostrar un botón para cada fun-

ción de una aplicación acabaríamos con un interfaz completamente abarrotado.

Los menús disponen de una serie de partes distintas. La primera es la barra de herramientas. Éste es el espacio en la parte superior de la ventana de la aplicación que agrupa los distintos menús. Dentro de este espacio tenemos distintos objetos que se refieren a distintas categorías de opciones. Finalmente, podemos pulsar dentro de esas categorías y podemos mostrar el menú emergente completo con una serie de opciones que están relacionadas con ese menú.

Si pulsamos en el menú Archivo de un programa normalmente nos encon-

tramos con opciones como Nuevo, Abrir, Cerrar, Guardar o Guardar como entre otros. Mientras que los usuarios sepan el tipo de función que necesitan deben de ser capaces de seleccionar el menú adecuado y acceder a la opción deseada.

Siempre debemos tener una idea clara respecto al tipo de funcionalidad que debe ir en cada categoría del menú y asegurarnos de que nuestros menús son lo suficientemente intuitivos como para persuadir al usuario de pulsar en el menú correcto para acceder a la opción deseada.

La organización de menús cada vez tiene manos en cuenta su uso y la consecuencia directa de esto es la convivencia de opciones no relacionadas en el mismo



menú. Un ejemplo puede ser la opción Work Offline (Trabajar desconectado) en el menú Archivo del programa Mozilla Firefox.

Otro ejemplo en Microsoft Windows es la aparición de la opción de apagado en el menú de Inicio. Alguno de vosotros no estaréis de acuerdo con estos ejemplos, pero todos debemos prestar atención al uso por parte del usuario de los menús al crearlos.

Creación de Nuestro Primer Menú

Para comenzar con nuestro primer menú necesitaremos crear el código base normal que aparece en casi todos los archivos XUL. Usaremos el código que aparece en el listado 1.

Ahora podemos crear la estructura de nuestro menú. El primer elemento que necesitamos crear es el que posiciona nuestro menú en la parte superior de la pantalla. Esto es llamado caja de herramientas y lo añadimos con la etiqueta `< toolbox >` (caja de herramientas):

```
<toolbox>
```

Como otros contenedores de gestión de geometría, la etiqueta "toolbox" posiciona el código en una parte específica de la pantalla. Otra opción es la de poder añadir el control especial "agarradera" de la barra de herramientas de forma que pueda ser separada de la ventana y ser movida en una ventana separada sobrepuesta. No usaremos esta opción para nuestros menús por

Otros lenguajes y XUL

Una de las mayores preguntas que se realiza al mirar a XUL es si hay otros lenguajes disponibles para escribir código XUL. Actualmente el único lenguaje soportado es Javascript, si bien se está pensando en la inclusión de otros lenguajes en versiones futuras de Mozilla. Entre estos lenguajes puede que estén incluidos Python o C#. Muchas de las discusiones entre los desarrolladores se giran alrededor de la creación de un nuevo nivel de funcionalidad de la aplicación Mozilla: la máquina virtual de Mozilla. Podemos encontrar más información al respecto del progreso del proyecto Mozilla en las Webs Mozillazine (<http://www.mozillazine.org/>) y en Planet Mozilla (<http://planet.mozilla.org/>).

ahora (se usa normalmente en barras de herramientas).

Lo siguiente que debemos hacer es crear una barra de menú en las que poner las entradas de nuestro menú. Nos referimos a la típica zona gris sobre la que se superponen los componentes de los menús. Podemos añadir esta zona usando la etiqueta "menubar" (barra de menú):

```
<menubar id="menubar">
```

Hemos especificado una identidad a esta barra de menú, práctica común cuando manejamos menús y botones en XUL.

Ahora estamos listos para agregar componentes a nuestro menú. Con este fin utilizamos la etiqueta "menu" (menú) con su correspondiente identidad para el atributo y nombre con la etiqueta del atributo:

```
<menu id="filemenu"
label="Archivo">
```

Ahora necesitamos llenar el menú con componentes. Para hacer esto necesitamos añadir el primer menú especial emergente usando la etiqueta `< menupopup >`:

```
<menupopup id="filepopup">
```

con nuestro menú emergente añadido procedemos a insertar objetos usando la etiqueta `< menuitem >` (elemento de menú) para cada uno de los distintos objetos:

```
<menuitem label="Nuevo"/>
<menuitem label="Abrir"/>
<menuitem label="Guardar"/>
```

Puede que hayas notado que no hay un atributo de identificación para cada menú. Esto es debido a que vamos a usar un método diferente para gestionar que elemento del menú se pulsa que veremos más adelante.

Cuando estemos creando nuestros menús puede que deseemos separarlos en diferentes secciones. Podemos hacerlo usando separadores de menús con la etiqueta `< menuseparator >` (Separador de menús):

```
<menuseparator/>
```

El elemento final puede ser la típica opción de cerrar tras el separador:

```
<menuitem label="Salir"/>
```

Para finalizar añadiremos todas la etiquetas de cierre que completan la estructura de nuestro menú:

```
</menupopup>
</menu>
</menubar>
</toolbox>
</window>
```

Nuestro archivo completo hasta ahora se muestra en el listado 2 y en la figura 1 vemos la estructura completada.

Más Funcionalidades de menú

Una de las funcionalidades más habituales en un menú es un sub-menú. Este tipo de menús especiales aparece cuando movemos el curso sobre un componente del menú principal.

Para crear un sub-menú debemos editar parte del código existente y añadir otra estructura para el mismo. Añadiremos un sub-menú al elemento Nuevo de nuestro menú, necesitando cambiar la siguiente línea:

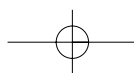
```
<menuitem label="Nuevo"/>
```

Cuando añadimos un sub-menú emergente debemos cambiar el elemento padre del sub-menú, modificando la etiqueta original `< menuitem >`, por la etiqueta `< menu >`:

```
<menu id="newmenu"
label="Nuevo">
```

Listado 1: first.xul

```
01 <?xml version="1.0"?>
02 <xml-stylesheet
   href="chrome://global/skin/"
   type="text/css"?>
03 <window
04   id="test-window"
05   title="Test Program"
06
   xmlns="http://www.mozilla.org/
   keymaster/gatekeeper/there.is.
   only.xul">
```



Ya hemos creado una estructura nueva. Ahora estamos listos para añadir el sub-menú emergente:

```
<menupopup id="secondpopup">
<menuitem
label="Plantilla"/>
<menuitem label="Fichero"/>
</menupopup>
```

Debemos recordar añadir las etiquetas de cierre:

```
</menu>
```

Ahora debemos ver algo similar a lo que se muestra en la figura 2.

Construcción de un Interfaz Completo

XUL nos muestra su potencial real cuando comenzamos a unir diferentes componentes hasta completar un interfaz.

Vamos a crear un interfaz completo con XUL que puede utilizarse como base de futuras interfaces. Construiremos este interfaz desde el principio como si de una herramienta de gestión de información se tratase. Esto involucrará la dedicación de determinadas partes de la pantalla a diferentes funciones y usos. Si bien no escribiremos la funcionalidad del interfaz, sí lo usaremos como base para escribir una aplicación completa en XUL.

Para hacer que sea más fácil de entender repasaremos cada línea del código individualmente y luego presentaremos el listado del código completo al final del artículo.

Comenzaremos añadiendo el código base XUL del listado 1.

Este código simplemente especifica a Mozilla que se trata de XUL y luego usa la etiqueta `<window>` (ventana) para crear la ventana padre. Una vez completado este código debemos especificar el archivo que contiene el Javascript que proporciona funcionalidad a nuestra interfaz:

```
<script src="code.js"/>
```

En nuestra interfaz ejemplo no estamos realmente escribiendo ningún código en JavaScript y solo nos concentraremos en XUL. Si estuviésemos escribiendo una aplicación XUL funcional

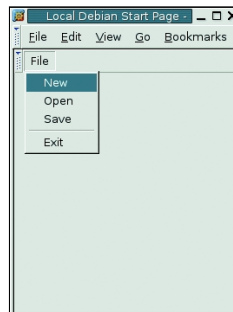


Figura 1: Un simple menú Archivo con una línea de separación incluida.

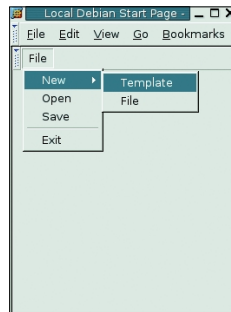


Figura 2: Añadir un sub-menú al menú principal en XUL es simple.

deberíamos asegurarnos de que la etiqueta con la que especificamos el código JavaScript es la primera etiqueta dentro de la etiqueta bloque `<window>`.

La primera parte de nuestro interfaz a crear es la estructura del menú. Especificaremos la etiqueta `<toolbox>` (caja de herramientas) para indicar que nuestro menú está posicionado en la parte superior de nuestra ventana. Esto nos asegura que nuestros menús y barras de herramientas (las veremos más adelante) se agrupan de forma conjunta en la caja de herramientas:

```
<toolbox>
```

Ahora podemos crear nuestra barra de menú principal y poner un menú y un sub-menú. Debemos recordar que cada menú se especifica con una etiqueta `<menu>` (esta etiqueta realmente añade el nombre del menú a la barra de menús), pero también debemos crear el menú emergente propiamente dicho (el menú que aparece cuando pulsamos sobre un elemento). Éste es el código:

```
<menubar id="samplemenubar">
<menu id="filemenu">
label="File">
<menupopup id="filepopup">
```

En este punto no tenemos ningún elemento en el menú emergente. El primer elemento, *Nuevo*, es un poco diferente porque es en sí un menú emergente. Para añadir este sub-menú creamos nuestras etiquetas `<menu>` y `<sub-menu>` y luego añadimos las etiquetas `<menuitem>` (elementos del menú). Es importante saber que si bien las otras

entradas del menú Archivo se añaden con la etiqueta `<menuitem>`, el objeto Nuevo es añadido con la etiqueta `<menu>` al ser un sub-menú:

```
<menu id="newmenu">
label="Nuevo">
<menupopup
id="secondpopup">
<menuitem
label="Plantilla"/>
<menuitem
label="Fichero"/>
</menupopup>
</menu>
```

Tras completar el menú Nuevo y su sub-menú podemos proceder a añadir el resto de nuestro componentes al menú Archivo. Hacemos esto con una serie de etiquetas `<menuitem>`, usando la etiqueta `<menuseparator>` para crear una línea entre las entradas de los elementos:

```
<menuitem label="Abrir"/>
<menuitem label="Guardar"/>
```

Listado 2: second.xul

```
01 <?xml version="1.0"?>
02 <?xml-stylesheet
href="chrome://global/skin/"
type="text/css"?>
03 <window
04 id="test-window"
05 title="Test Program"
06
xmlns="http://www.mozilla.org/
keymaster/gatekeeper/there.is.
only.xul">
07 <toolbox>
08 <menubar id="menubar">
09 <menu id="filemenu"
label="Fichero">
10 <menupopup id="filepopup">
11 <menuitem label="Nuevo"/>
12 <menuitem label="Abrir"/>
13 <menuitem label="Guardar"/>
14 <menuseparator/>
15 <menuitem label="Salir"/>
16 </menupopup>
17 </menu>
18 </menubar>
19 </toolbox>
20 </window>
```



```
<menuseparator/>
<menuitem label="Salir"/>
</menupopup>
</menu>
```

Ahora procedemos a añadir el segundo menú a la barra de herramientas. Éste será el menú de edición (Edit), usando de nuevo las etiquetas `<menu>`, `<menupopup>`, y `<menuitem>`:

```
<menu id="editmenu"
label="Edit">
  <menupopup id="editpopup">
    <menuitem label="Deshacer"/>
    <menuitem label="Rehacer"/>
  </menupopup>
</menu>
```

Para finalizar nuestra estructura, necesitamos añadir las etiquetas de cierre `<menubar>` y `<toolbox>`:

```
</menubar>
</toolbox>
```

La parte principal de nuestra interfaz va a estar compuesta, a la izquierda, por una combinación de una caja combo, una caja de listado y una barra de división reajutable, y a la derecha el área principal de texto y algunos otros botones. Con esto en mente debemos tener algunos componentes encima de otros (la caja combo está encima de la caja de listado, por ejemplo) y algunos otros componentes al lado de otros (por ejemplo botones). Esto implica la combinación de las etiquetas `<hbox>` y `<vbox>` con el fin de agrupar componentes de determinadas formas.

Comenzaremos usando la etiqueta `<box>` (caja) para indicar que estamos usando un diseño tipo caja para todos nuestros componentes, y luego creamos la etiqueta `<vbox>` para disponer la caja combo y la caja de listado una encima de la otra. Cada uno de los componentes usa `<flex=1>` para extender el propio componente y ocupar todo el espacio disponible. Esta

coletilla se utiliza a lo largo de todo el código.

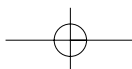
```
<box flex="1">
  <vbox flex="1">
```

Nuestro primeros componentes van a ser una etiqueta y una caja combo, los cuales van a estar agrupados por la etiqueta `<hbox>` para que se coloquen uno al lado del otro. Primero creamos la etiqueta `<label>` con el modo texto y luego creamos la caja combo (es como se denomina en XUL una lista de menú) con la etiqueta `<menulist>`. Dentro de esta etiqueta creamos una serie de entradas `<menuitem>` para cada objeto en la caja. También hemos dispuesto el atributo `editable="false"` para la lista de nuestro menú con la intención de que los usuarios no puedan cambiar los contenidos de la caja:

```
@LI <hbox> <label value="Mode"/>
  <menulist editable="false"
flex="1"> <menupopup> <menu-
```

Listado 3: third.xul Completo

```
01 <?xml version="1.0"?>
02 <?xml-stylesheet
  href="chrome://global/skin/"
  type="text/css"?>
03 <window
04 id="testwindow"
05 title="XUL Interface"
06
  xmlns="http://www.mozilla.org/
  keymaster/gatekeeper/there.is.
  only.xul">
07 <script src="code.js"/>
08 <toolbox>
09 <menubar id="samplemenubar">
10 <menu id="filemenu"
  label="Fichero">
11 <menupopup id="filepopup">
12 <menu id="newmenu"
  label="Nuevo">
13 <menupopup id="secondpopup">
14 <menuitem
  label="Plantilla"/>
15 <menuitem label="Fichero"/>
16 </menupopup>
17
18 </menu>
19 <menuitem label="Abrir"/>
20 <menuitem label="Guardar"/>
21 <menuseparator/>
22 <menuitem label="Salir"/>
23 </menupopup>
24 </menu>
25 <menu id="editmenu"
  label="Editar">
26 <menupopup id="editpopup">
27 <menuitem label="Deshacer"/>
28 <menuitem label="Rehacer"/>
29 </menupopup>
30 </menu>
31 </menubar>
32 </toolbox>
33 <box flex="1">
34 <vbox flex="1">
35 <hbox>
36 <label value="Mode"/>
37 <menulist editable="false"
  flex="1">
38 <menupopup>
39 <menuitem label="Simple"/>
40 <menuitem label="Avanzado"/>
41 <menuitem label="Experto"/>
42 </menupopup>
43 </menulist>
44 </hbox>
45 <listbox>
46 <listitem label="Facturas"/>
47 <listitem label="Gastos"/>
48 <listitem label="Pequeños
  Gastos"/>
49 <listitem label="Otro"/>
50 </listbox>
51 </vbox>
52 <splitter collapse="before">
53 <grippy/>
54 </splitter>
55 <vbox flex="1">
56 <hbox>
57
58 <button id="addbutton"
  label="Añadir"/>
59 <button id="deletebutton"
  label="Eliminar"/>
60 <button id="editbutton"
  label="Editar"/>
61 </hbox>
62 <textbox id="posttext"
  multiline="true" flex="1"/>
63 <hbox>
64 <button id="findbutton"
  label="Previsualizar"
  default="true"/>
65 <button id="cancelbutton"
  label="Publicar"/>
66 </hbox>
67 </vbox>
68 </box>
69 </window>
```



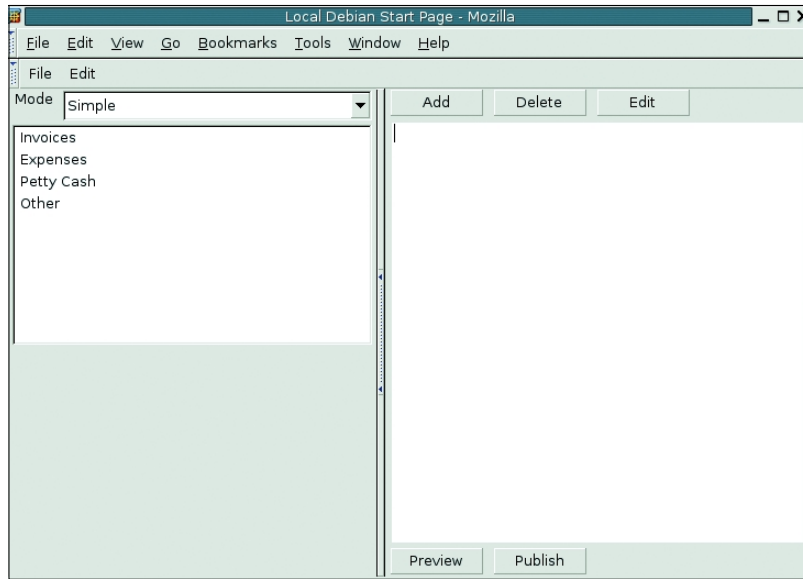


Figura 3: Un interfaz XUL completo.

item label= "Simple"/> <menuitem label= "Avanzado"/> <menuitem label= "Experto"/> </menupopup > </menulist > </hbox >

El componente que falta por añadir es la caja de listado. Para crearlo usamos la etiqueta <listbox> y añadimos cada objeto a la etiqueta <listitem > :

```
<listbox>
<listitem label= "Factura"/>
<listitem label= "Gastos"/>
<listitem label= "Pequeños
Gastos"/>
<listitem label= "Otros"/>
</listbox>
```

Con anterioridad abrimos la etiqueta <vbox> en el código para disponer las cajas combo y de listado una encima de la otra. Ahora debemos cerrar esta etiqueta usando la etiqueta de cierre <vbox > :

```
</vbox>
```

Hasta ahora hemos creado componentes que aparecen en la parte izquierda de nuestro interfaz. En casos en los que claramente separemos un lado del interfaz del otro debemos usar la etiqueta <splitter > para tener un separador ajustable entre las dos zonas. Una de las posibilidades de esta etiqueta es que podemos colapsar un lado y especi-

camos el atributo <collapse > (colapsar) para especificar los componentes antes de la etiqueta en el código (nuestra caja combo y listado). También hemos añadido la etiqueta <grippy > (agarradera) para proporcionar una manera de mover el separador:

```
<splitter collapse="before">
<grippy/>
</splitter>
```

Para componentes situados a la derecha del separador vamos a agrupar grupos de componentes unos encima de otros. Primero creamos la etiqueta <vbox > :

```
<vbox flex="1">
```

Nuestro primer grupo de componentes esta formado por los tres botones que aparecen cerca del marco superior de la ventana. Creamos la etiqueta "hbox" para disponerlos unos al lado de los otros y luego añadimos la etiqueta "button":

```
<hbox>
<button id="addbutton"
label="Añadir"/>
<button id="deletebutton"
label="Eliminar"/>
<button id="editbutton"
label="Editar"/>
</hbox>
```

El siguiente componente es un componente de entrada de datos de mucho texto. Éste es el tipo de componente que se utilizaría para introducir texto en un editor de textos, por lo que nos debemos asegurar de que el componente es capaz de soportar más de una línea de texto. Podemos hacer esto con el atributo *multilinea*:

```
<textbox id="posttext"
multiline="true" flex="1"/>
```

Nuestro conjunto final de componentes es una colección de botones que funcionan de la misma forma que los botones Añadir, Eliminar y Editar que hemos creado con anterioridad:

```
<hbox>
<button id="findbutton" label="
Previsualización"
default="true"/>
<button id="cancelbutton"
label="Publicar"/>
</hbox>
```

Para completar nuestro código ahora debemos añadir la etiquetas de cierre <vbox >, <box > y <window > :

```
</vbox>
</box>
</window>
```

El listado 3 es el código completo. Cuando ejecutemos este código debemos ver algo parecido a lo que se muestra en la figura 3.

El Siguiente Paso

En este número hemos agrupado todos nuestro conocimientos actuales de construcción de interfaces en XUL. A medida que continuemos explorando XUL veremos las funcionalidades de aplicaciones XUL en más detalle, incluyendo el uso de PHP para conseguir archivos XUL más dinámicos.

Hasta que continuemos el próximo mes debemos seguir jugando con las diferentes etiquetas de interfaz de XUL y experimentando con distintas combinaciones de componentes. Cuantas más experiencia tengamos mejor vamos a ser capaces de resolver problemas. ■