

El WidgetServer.

El programa WidgetServer sirve para realizar interfaces gráficas usando un lenguaje muy sencillo. Los programas pueden interactuar con el WidgetServer a través de la entrada estándar y obtener la salida de los eventos por la salida estándar.

Esto hace que sea independiente del lenguaje de programación, pudiendo crear ventanas tanto desde de C, como Python, Java, Perl, Bash,...

Un ejemplo rápido.

Comenzaremos por un ejemplo sencillo:

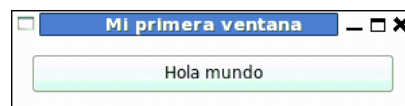
```
<window:w1>
  <title:Mi primera ventana/>
  <button:b1>
    <text>
      Hola mundo
    </text>
  </button>
</window>
```

Importante: Se debe escribir cada comando en una nueva línea y dejar, al menos, una línea en blanco al final del archivo.

Para ver la ventana que se genera se le debe pasar por la entrada estándar los comandos anteriores al WidgetServer, o bien, guardarlos en un archivo, por ejemplo, *ej1.ws* y pasárselos al servidor con el comando:

```
cat ej1.ws|widgetserver
```

La salida que se obtiene es similar a:



Este ejemplo sólo nos muestra la ventana con un botón, pero no se puede interaccionar con ella.

Para poder capturar los eventos que se generan al pulsar el botón se añadiría el código:

```
<window:w1>
  <title:Mi primera ventana/>
  <button:b1>
    <text>
      Hola mundo
    </text>
    <listen:clicked/>
  </button>
</window>
```

Con el comando añadido, lo que se le está diciendo al servidor es que se desean escuchar los eventos de hacer click con el ratón sobre el botón. Si este texto lo introducimos en un archivo, por ejemplo, *ej2.ws* y se le pasa al servidor con el comando:

```
cat ej2.ws|widgetserver
```

Se obtiene una ventana similar a la anterior pero ahora, al pulsar el botón se generarán unos mensajes como los siguientes:

*clicked: b1

Esto nos indica que se ha hecho click con el ratón sobre el botón con etiqueta b1.

Los mensajes que generan los eventos siempre son de la forma:

**Nombre del evento: nombre del widget*

Tags y Widgets.

En html a cada elemento de la forma <texto> se le denomina un tag. Los tags del WidgetServer no siguen la sintaxis de html pero es muy similar.

A cada objeto que va a mostrar el WidgetServer se le denomina Widget. Los widgets se representan por un par de tags de la forma:

```
<tipo de widget:nombre del widget>
...
</tipo de widget>
```

Entre ambos tags se introducirán las propiedades del widget. Por ejemplo:

```
<window:w1>
    <title:Mi primera ventana/>
</window>
```

Indica que se crearán un widget del tipo window, con nombre w1 y su título será “Mi primera ventana”.

Los tags que no van a contener nada en su interior deben ser de la forma:

```
<texto del tag/>
```

Como el tag title del widget window.

La línea de comandos.

Para ejecutar el WidgetServer se debe usar el comando *widgetserver*. Con la opción *--help* mostrará la siguiente ayuda:

```
widgetserver [--help] [--debug] [-exec path]
Starts WidgetServer
--help Shows this help
--debug shows debug output
-exec path Executes path in WidgetServer. All input and output are redirected by
WidgetServer
```

La opción *--help* muestra la ayuda.

La opción *-debug* muestra una salida que puede ser útil para depurar las ventanas.

La opción *-exec path* Ejecuta el programa que se encuentra en path y redirecciona su entrada y salida estándar al WidgetServer. Es útil a la hora de programar, pues los programas sólo tienen que enviar los comandos del WidgetServer por la salida estándar y obtienen las respuestas del servidor por la entrada estándar.

Si se ejecuta el comando *widgetserver* sin opciones espera a recibir órdenes por la entrada estándar.

Tipos de widgets.

Se van a ver ahora algunos tipos de widgets.

Widget: Label.

El widget label sirve para mostrar texto dentro de una ventana su sintaxis es:

```
<label:nombre del widget>  
Texto a mostrar  
</label>
```

Por ejemplo:

```
<window:w1>  
  <label:l1>  
    Un original mensaje  
  </label>  
  <label:l2>  
    <html>  
      <b>¡¡Sí, admite HTML!!</b>  
    </html>  
  </label>  
</window>
```

Como se puede apreciar en el ejemplo, se le puede introducir texto, o bien, html añadiendo los tags <html> </html>. La ventana que se genera será:



Widget: Window.

El widget window sirve para mostrar una ventana. Su sintaxis es:

```
<window:nombre del widget>  
Widgets y propiedades  
</window>
```

Entre ambos tags se introducen los widgets que se desea que contenga la ventana y las propiedades de la ventana.

Las propiedades de la ventana son:

```
<title: Título/>
```

Selecciona el título de la ventana.

```
<icon:path/>
```

Selecciona el icono de la ventana. El icono se encuentra en el path indicado.

```
<full screen: true/>
```

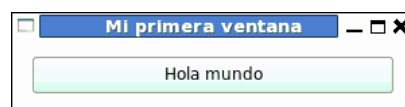
Pone la ventana con tamaño máximo y sin decoraciones.

```
<full screen: false/>
```

Pone la ventana con tamaño normal.

Widget: Button.

Muestra un botón con un texto y/o un icono.



Su sintaxis:

```
<button:nombre del widget>
```

```
...Propiedades...
</button>
```

Propiedades:

```
<text>
    ...Texto...
</text>
```

Inserta el texto en el widget, eliminado los espacios en blanco.

```
<text end="cadena fin">
    ...Texto...
</text end="cadena fin">
```

Inserta el texto en el widget, sin eliminar los espacios en blanco.

```
<icon:path/>
```

Selecciona el icono del botón. El icono se encuentra en el path indicado.

```
<listen:clicked/>
```

Indica que se desean escuchar el evento click del ratón sobre el botón.

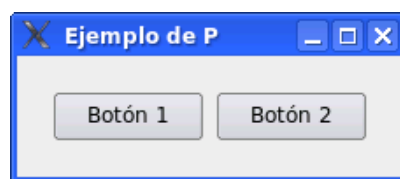
Las propiedades icon y listen son opcionales.

Ejemplo:

```
<window:w1>
    <title:Mi primera ventana/>
    <button:b1>
        <text>
            Hola mundo
        </text>
        <listen:clicked/>
        <icon:bookmark.png/>
    </button>
</window>
```

Widget: P.

Representa un párrafo en el que se van introduciendo los widgets en horizontal.



Sintaxis:

```
<p>
...Widgets...
</p>
```

Ejemplo:

```
<window:w1>
    <title:Ejemplo de P/>
    <p>
        <button:b1>
            <text>
                Botón 1
            </text>
        </button>
        <button:b2>
            <text>
                Botón 2
            </text>
        </button>
    </p>
</window>
```

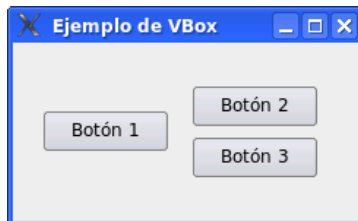
```

        </text>
    </button>
</p>
</window>

```

Widget: VBox.

Representa una pila en la que se van introduciendo los widgets en vertical.



Sintaxis:

```

<vbox>
...Widgets...
</vbox>

```

Ejemplo:

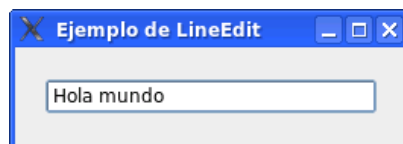
```

<window:w1>
    <title:Ejemplo de VBox/>
    <p>
        <button:b1>
            <text>
                Botón 1
            </text>
        </button>
        <vbox>
            <button:b2>
                <text>
                    Botón 2
                </text>
            </button>
            <button:b3>
                <text>
                    Botón 3
                </text>
            </button>
        </vbox>
    </p>
</window>

```

Widget: LineEdit.

Representa una caja en la que se puede editar una línea de texto.



Sintaxis:

```

<lineedit: nombre del widget>
...Propiedades...
</lineEdit>

```

Las propiedades son:

```

<text: Texto />

```

Selecciona el texto del lineedit.

<getText/>

Obtiene el texto del widget.

Ejemplo:

```
<window:w1>
  <title:Ejemplo de LineEdit/>
  <p>
    <lineedit:le1>
      <text:Hola mundo/>
    </lineedit>
  </p>
</window>
```

Widget: TextEdit.

Representa una caja en la que se puede editar una línea de texto.



Sintaxis:

```
<textedit: nombre del widget>
  ...Propiedades...
</textfield>
```

Las propiedades son:

```
<text>
  ...Texto...
</text>
```

Inserta el texto en el textedit, eliminando los espacios en blanco.

```
<text end="cadena fin">
  ...Texto...
</text end="cadena fin">
```

Inserta el texto en el textedit, sin eliminar los espacios en blanco.

<getText/>

Obtiene el texto del widget. Devuelve, primero, un número que representa el número de bytes que se van a devolver. Después se coloca el texto del widget.

Ejemplo:

```
<window:w1>
  <title:Ejemplo de TextEdit/>
  <p>
    <textedit:le1>
      <text>
        Hola mundo.
      </text>
    </textfield>
```

```
</p>
</window>
```

Widget: HTML.

Representa un sencillo navegador html.



Sintaxis:

```
<html: nombre del widget>
...html...
</html>
```

El html que se puede representar es muy sencillo. En el código html que se introduce, no es necesario que cumpla la sintaxis de las ventanas del widgetserver, es decir, no es necesario colocar un tag por línea.

Ejemplo:

```
<window:w1>
  <title:Ejemplo de Html/>
  <html:html1>
    <b>Esto es un ejemplo de html.</b><br>
    <center></center>
  </html>
</window>
```

Widget: List.

Muestra listas o árboles de textos.

A screenshot of a window titled "Ejemplo List". Inside the window, there is a table with 3 columns and 3 rows of numbers. The first row has headers "Columna 1", "Columna 2", and "Columna 3". The second row has "1", "2", and "3". The third row has "4", "5", and "6". The fourth row has "7", "8", and "9". There is a small square icon to the left of the number 4.

Columna 1	Columna 2	Columna 3
1	2	3
4	5	6
7	8	9

Su sintaxis es la siguiente:

```
<list: nombre del widget>
  ...Propiedades del widget...
  ..items...
</list>
```

Propiedades:

```
<selection mode:multiple/>
```

Permite seleccionar varias filas a la vez.

```
<selection mode:simple/>
```

Sólo permite seleccionar una fila.

```
<listen:selection changed/>
```

Escucha los eventos de selección de las filas.

Los items son las filas de texto que se van a mostrar:

```
<item:nombre del item>
  ...Filas con los datos. Una fila por cada columna...
```

```
...Otros items
</item>
```

Las cabeceras son los títulos de las columnas que se van a mostrar:

```
<headers>
...Filas con los nombres de las columnas...
</headers>
```

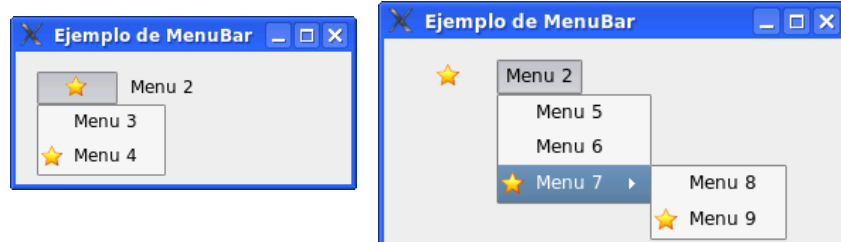
Por ejemplo:

```
<window:w1>
  <title:Ejemplo List/>
  <list:l1>
    <listen:selection changed/>
    <selection mode:multiple/>
    <headers>
      Columna 1
      Columna 2
      Columna 3
    </headers>
    <item:i1>
      1
      2
      3
    </item>
    <item:i2>
      4
      5
      6
      <item:i3>
        7
        8
        9
      </item>
    </item>
  </list>
</window>
```

Ver el apartado “Un ejemplo completo” para ver otro ejemplo de este widget.

Widget: MenuBar.

Crea una barra de menú con menús.



Su sintaxis:

```
<menubar: nombre del widget>
..items...
</menubar>
```

Los items son los menús que se van a mostrar. Hay 2 tipos de items:

- Los que no tienen hijos:

Su sintaxis es:

```
<item:nombre_del_item text="Texto del menú" icon="path"/>
```


El campo icon es opcional.

- Los que tienen hijos:

```
<item:nombre_del_item text="Texto del menú" icon="path">
    ...items...
</item>
```

El campo icon es opcional.

Un ejemplo de este widget:

```
<window:w1>
  <title:Ejemplo de MenuBar/>
  <menubar:menubar1>
    <item:m1 text="Menu 1" icon="bookmark.png">
      <item:m3 text="Menu 3"/>
      <item:m4 text="Menu 4" icon="bookmark.png"/>
    </item>
    <item:m2 text="Menu 2">
      <item:m5 text="Menu 5"/>
      <item:m6 text="Menu 6"/>
      <item:m7 text="Menu 7" icon="bookmark.png">
        <item:m8 text="Menu 8"/>
        <item:m9 text="Menu 9" icon="bookmark.png"/>
      </item>
    </item>
  </menubar>
</window>
```

Ver el apartado “Un ejemplo completo” para ver otro ejemplo de este widget.

Widget: Workspace.

El widget workspace sirve para mostrar un área con ventanas en su interior. Su sintaxis es:

```
<workspace:nombre del widget>
Widgets y propiedades
</workspace>
```

Entre ambos tags se introducen los widgets que se desea que contenga.

Comandos generales.

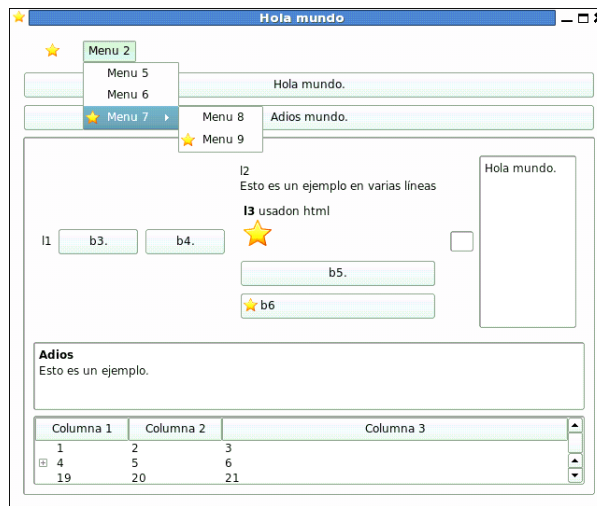
Existen algunos comandos que pueden ser de utilidad:

```
<quit/>
```

Sale del WidgetServer.

Un ejemplo completo.

Veamos ahora un ejemplo más completo. Este ejemplo generará una ventana como la siguiente:



El código que la genera es:

```
<window:w>
  <title: Hola mundo/>
  <icon:bookmark.png/>
  <menubar:menubar1>
    <item:m1 text="Menu 1" icon="bookmark.png">
      <item:m3 text="Menu 3"/>
      <item:m4 text="Menu 4" icon="bookmark.png"/>
    </item>
    <item:m2 text="Menu 2">
      <item:m5 text="Menu 5"/>
      <item:m6 text="Menu 6"/>
      <item:m7 text="Menu 7" icon="bookmark.png">
        <item:m8 text="Menu 8"/>
        <item:m9 text="Menu 9" icon="bookmark.png"/>
      </item>
    </item>
  </menubar>
  <button:b1>
    <text>
      Hola mundo.
    </text>
  </button>
  <button:b2>
    <text>
      Adios mundo.
    </text>
  </button>
  <window:scroll>
  <p>
    <label:l1>
      l1
    </label>
    <button:b3>
      <text>
        b3.
      </text>
    </button>
    <button:b4>
      <text>
        b4.
      </text>
    </button>
    <vbox>
      <label:l2>
        l2
```

```

        Esto es un ejemplo en varias líneas
    </label>
    <label:l3>
        <html>
            <b>l3</b> usadon html<br>
            
        </html>
    </label>
    <button:b5>
        <text>
            b5.
        </text>
    </button>
    <button:b6>
        <text>
            b6
        </text>
        <icon:bookmark.png/>
        <listen:clicked/>
    </button>
</vbox>
<lineedit:l1>
<text:Hola mundo/>
</lineedit>
<textedit:t1>
    <text>
        Hola mundo.
    </text>
</textedit>
</p>
<html:html1>
    <b>Hola mundo</b><br>
    Esto es un ejemplo.
</html>
<list:list1>
    <listen:selection changed/>
    <selection mode:multiple/>
    <headers>
        Columna 1
        Columna 2
        Columna 3
    </headers>
    <item:pepe>
        1
        2
        3
    </item>
    <item>
        4
        5
        6
    <item>
        7
        8
        9
    </item>
    <item>
        10
        11
        12
    </item>
    <item>
        13
        14

```

```
15
    <item>
16
17
18
    </item>
    </item>
    </item>
    <item>
19
20
21
    </item>
</list>
</window>
</window>

<lineedit:le1>
    <getText/>
</lineEdit>

<html:html1>
    <b>Adios</b><br>
    Esto es un ejemplo.
</html>
```