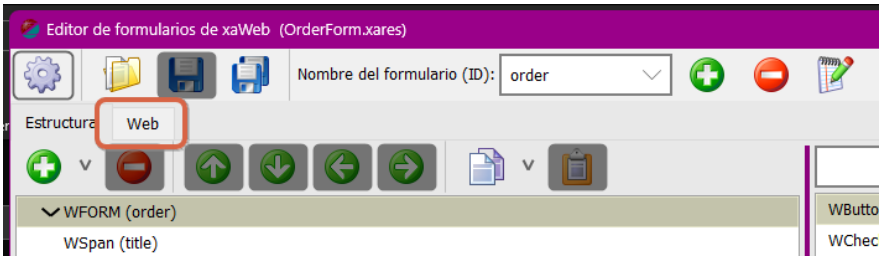
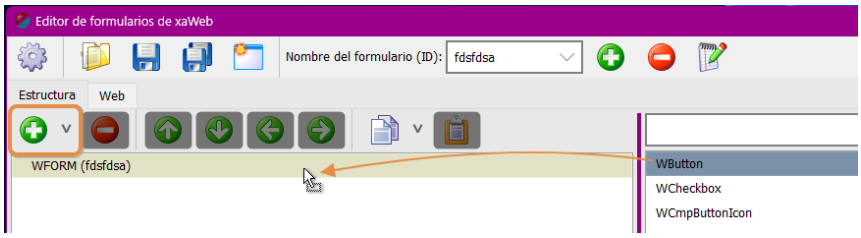
Por Ignacio Ortiz de Zúñiga

Editor de formularios

xaWeb incorpora una herramienta para la creación visual de formularios. Esta herramienta facilita enormemente la creación de formularios (elementos <form></form>) en xaWeb. El resultado final no es completamente WYSYWYG (‘*What you see is what you get*’), pero casi, ya que simplemente con cambiar de pestaña en la aplicación podemos ver el resultado final de nuestro formulario siendo la respuesta prácticamente inmediata.



El funcionamiento es muy sencillo. Desde la aplicación podrá crear todos los formularios que desee, pero sólo uno de ellos puede estar activo. En el formulario activo podrá incluir todos los controles clásicos tipo <input> que desee. Para ello puede utilizar la opción de arrastrar controles desde la ventana de controles accesibles, o bien utilizar el botón de añadir control que se representa con un símbolo ‘+’.

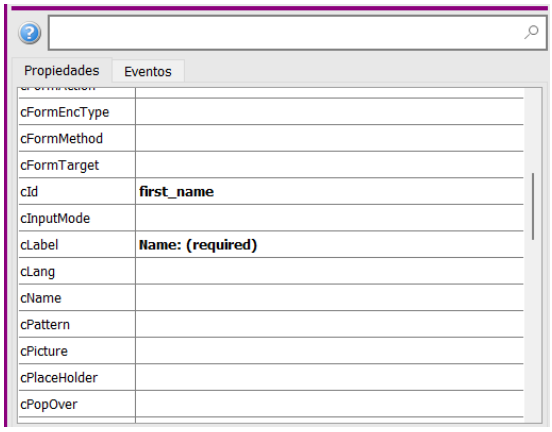


Dependiendo de donde deposite el control, éste formará parte en una posición específica del formulario sobre una clásica estructura de árbol. Los botones de navegación de color verde nos permiten modificar el orden de creación de cada control y su posición en el árbol que configura la estructura del formulario.

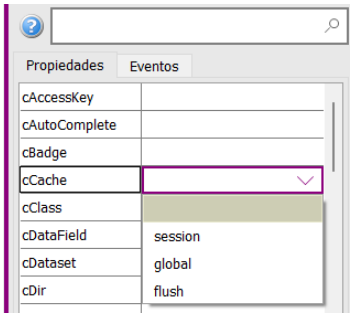
Esta sería una clásica estructura de árbol de un formulario:

Observe como el primer elemento del formulario es un control de tipo que incorpora el título de este. A Continuación, hay un control de tipo <fieldset> que engloba todos los controles tipo <input>, que en el ejemplo son controles de edición tipo <input-edit>, una lista desplegable tipo <select> y además una agrupación de controles tipo <input-radio> que se engloban en un control WRadiomenu. Por último, hay un control tipo WFlexRow, que no es más que un que valores de estilo tipo 'css-flex', que nos permitirán centrar los botones que depende de él en la pantalla.

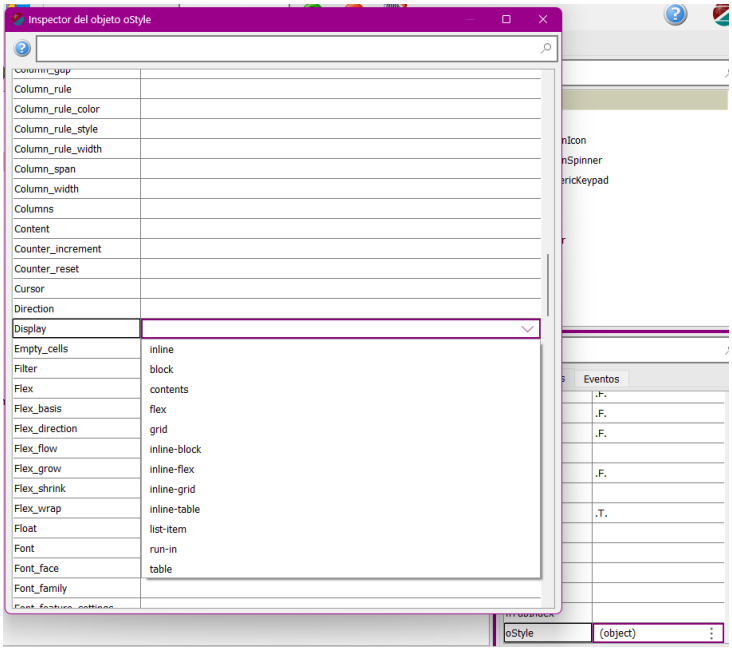
Todas las propiedades y eventos de estos controles pueden ser asignadas desde el inspector de objetos:



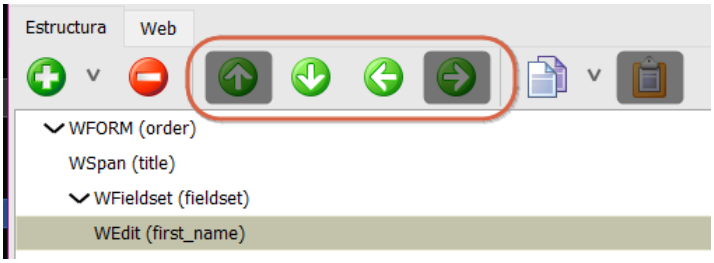
Cuando la propiedad admite una lista, ésta le será mostrada en el mismo inspector:



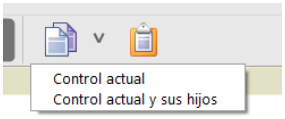
Cuando se trate de una matriz (la propiedad empieza por la letra 'a') se le mostrará un pequeño editor en cual podrá añadir tantos elementos como desee, separándolos cada uno de ellos con un retorno de carro (pulsación Intro). Un caso especial es la propiedad oStyle que tienen todos los controles. Al hacer clic sobre los tres puntos verticales que muestra la propiedad en su edición, se desplegará un nuevo formulario que soporta el 99% de los estilos posibles de CSS y que simplifica mucho su uso:



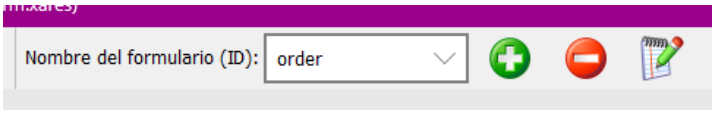
Una vez que ha depositado un control dentro del formulario es muy sencillo cambiar su posición en el árbol de éste. No sólo puede subir o bajar su posición en su contenedor, sino que incluso puede cambiar la rama del árbol en la que se encuentra. Utilice los botones de navegación para cambiar la posición de cualquier control:



Cualquier control o rama completa puede ser depositada en el portapapeles y para ello sólo deberá posicionarse en el control seleccionado y pulsar el botón de copiar al portapapeles. El cual le da elegir entre copiar el control actual sólo o el control con toda su rama. A continuación, sólo deberá ir al control que desee y realizar una operación de pegado, pulsando el botón de pegado de la barra de botones. Es factible realizar operaciones de copiado desde un formulario a otro formulario. Si realiza operaciones de copiado en un mismo formulario, deberá modificar los IDs de todos los controles involucrados en el proceso.



Un único archivo de recursos puede tener un número ilimitado de formularios. La creación, selección, borrado y edición del nombre de cada uno de ellos se realiza desde esta sección:



- La lista desplegable le permite seleccionar el formulario activo
- El botón '+' le permite añadir un nuevo formulario al entorno
- El botón '-' se utiliza para borrar el formulario actual
- El botón de edición (cuaderno con lápiz) se utiliza para cambiar el nombre del formulario

El editor de formulario exige que cada control que se utilice tenga su propiedad cID establecida. El editor establece su valor de forma automática cuando es depositado en el formulario, pero usted puede cambiarlo si así lo desea. De hecho, es lo más recomendable si luego pretende acceder a dicho control desde código. El editor de formularios no controla si ha asignado un ID que ya existe en algún control, pero le adelanto, que, si fuese así, cuando intente mostrar el formulario se producirá un error por ID duplicado.

NOTA IMPORTANTE

El editor de formularios no controla si ha asignado un ID que ya existe en algún otro control. En el caso de se produzca un duplicado se producirá un error por ID duplicado al visualizar el formulario con la pestaña 'web'.

FUNCIONAMIENTO

Cualquier proyecto creado con el editor de formularios de xaWeb se guarda en un único archivo JSON. Un proyecto de extensión 'xares', que es la extensión usada por la aplicación, que puede

incluir múltiples formularios. La idea es que todos los formularios que utilice el CGI estén en un único archivo.

La clase WFormManager() será la encargada de cargar dicho archivo de recursos y desplegar cualquier formulario que tenga en su interior con una simple instrucción.

Para que el editor de formularios pueda desplegar rápidamente el formulario sin tener que ejecutar el CGI constantemente, utiliza la técnica que otorga el método Engine:Listen() a xaWeb que permite tener el CGI en modo latente, esperando conexiones en un puerto TCP/IP. Es por ello por lo que el editor solicita en su configuración la ubicación de los archivos CGI que harán el trabajo.

Cuando arranque la aplicación puede observar, como en segundo plano se está realizando alguna operación adicional (el ratón muestra además el símbolo de 'trabajando'). La operación que se está realizando es la carga del CGI. Este proceso puede tardar algunos segundos, y hasta que no termine la carga no podrá visualizar correctamente ningún formulario. En el caso de que lo intente, recibirá un mensaje de error. Si esto ocurre, sólo necesita volver a repetir la operación de cambio de pestaña.

Cuando cierra la aplicación, se cierra de forma automática el CGI. Incluso si se produce un error de ejecución en la aplicación, el CGI se descarga igualmente. Si vuelve a ejecutar la aplicación y recibe un error de que el puerto TCP/IP está ocupado, es probable que el CGI anterior no se haya descargado correctamente y deberá hacerlo usted de forma manual desde el administrador de tareas de Windows.

PUESTA EN MARCHA

La aplicación se encuentra en el directorio \bin de la instalación de xaWeb y durante el proceso de instalación de xaWeb se crea un enlace al mismo en el menú de aplicaciones. La primera operación que deberá será configurar el editor pulsando el botón de la rueda que le mostrará el siguiente diálogo:

Observe que hay dos modos de trabajo:

1. Modo estándar
2. Modo 'Materialize'

xaWeb se comporta de forma completamente distinta si se ha enlazado con Materialize o no. Muchos controles, como WButton, WEdit y otros de tipo <input> tienen distintas propiedades y métodos según el escenario. Por dicho motivo hay dos CGIs posibles a utilizar y debe de establecer el modo de trabajo antes de usar la aplicación. Cada de vez que cambie el modo de trabajo la aplicación necesita reinicializarse.

Otra consecuencia de este doble modo de trabajo es que los recursos creados en un determinado modo son incompatibles con el otro modo. Es decir, los recursos creados en modo estándar no pueden ser usados en aplicaciones que usen Materialize y viceversa. La clase WFormManager genera un error cuando intenta utilizar un recurso incompatible.

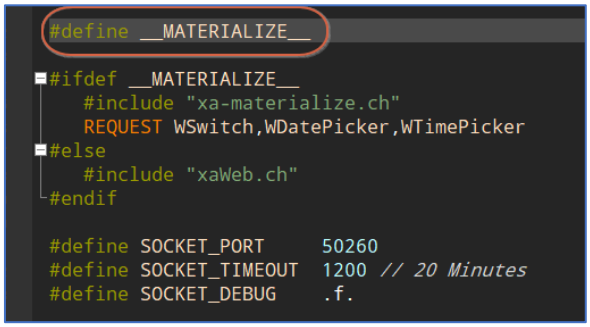
El puerto TCP/IP en principio no hay que cambiarlo, a no ser que la aplicación le indique que el puerto está ocupado.

El campo 'Document root' se refiere a la carpeta donde el editor va a guardar el archivo HTML que luego leerá el navegador web.

NOTA IMPORTANTE

Es necesario tener instalado el navegador web Edge de Microsoft para que funcione el sistema de visualización de los formularios.

Los CGIs que arranca de forma automática de la aplicación son muy básicos y los dos proyectos con todo su código fuente se encuentra en \xaWeb\samples\FormRunner. Es muy recomendable que modifique ambos proyectos a su antojo para incluir, por ejemplo, sus propias hojas de estilo (Css) o imágenes de fondo. Todos estos ficheros externos deberán estar disponibles en una subcarpeta de donde apunta el campo 'Document root' del formulario de configuración. Es muy importante que cuando enlace para Materialize deje el define __MATERIALIZE__ sin comentar y hago lo contrario cuando desee enlazar para la versión estándar del CGI.



UTILIZACIÓN DESDE XAWEB

xaWeb incorpora una clase de nombre **WFormManager** que recibe como único parámetro en su constructor del nombre del fichero de recursos a utilizar. El despliegue de cualquier formulario que se encuentre en el fichero de recursos se realiza con el método WFormManager:Deploy(<cForm>, <oDiv>), que recibe el nombre del formulario y el contenedor donde se va a desplegar. Eso es todo.

```
WITH OBJECT oDiv := WDiv():New( Self )
   :AddClass( "container" )
   WITH OBJECT WFormManager():New( cResfile )
      IF Empty( :cError )
         oFrm := :DeployForm( "", oDiv ) // Loads first form in resources
         IF !HB_IsObject( oFrm )
            ECHO "Form Manager Error: (" + :cError + ")" INTO oDiv
         ENDIF
      ELSE
         ECHO "Error on WFormManager constructor: (" + :cError + ")" INTO oDiv
      ENDIF
      :End()
   END WITH
END WITH
```

SOLUCIÓN DE PROBLEMAS

El principal problema que puede tener es que con la instanciación del CGI y la apertura del puerto TCP/IP, que como se ha explicado con anterioridad son de fácil solución, simplemente cambiando el puerto TCP/IP en la configuración del programa.

Si al cambiar a la pestaña 'Web' le muestra un error de socket no disponible, normalmente será porque ha no le ha dado tiempo al CGI a ponerse en modo escucha aún. Simplemente repita el proceso de cambio de pestaña para ver si se arregla.

Si no se muestra el formulario y recibe un error de que no existe el archivo HTML, es probable que el error esté en el propio proyecto xaWeb\samples\FormRunner si usted lo ha modificado o sea un error del propio xaWeb que aún no hemos conseguido localizar. En cualquier caso, le recomendamos que edite el proyecto xaWeb\samples\FormRunner y establezca el define de la propiedad SOCKET_DEBUG a verdadero de forma temporal.

```
#define __MATERIALIZE__

#ifdef __MATERIALIZE__
   #include "xa-materialize.ch"
   REQUEST WSwitch,WDatePicker,WTimePicker
#else
   #include "xaWeb.ch"
#endif

#define SOCKET_PORT     50260
#define SOCKET_TIMEOUT  1200 // 20 Minutes
#define SOCKET_DEBUG    .f.
```

Cuando haga esto, observará que se crea un archivo de nombre xaWeb.log con muchísima información de todas las operaciones que se realizan en el CGI. Es muy probable que dicho archivo le de la pista de lo que puede estar pasando. En caso contrario, póngase en contacto con OZ Software enviando dicho archivo y el proyecto FormRunner si lo tiene modificado.

CONTENIDO