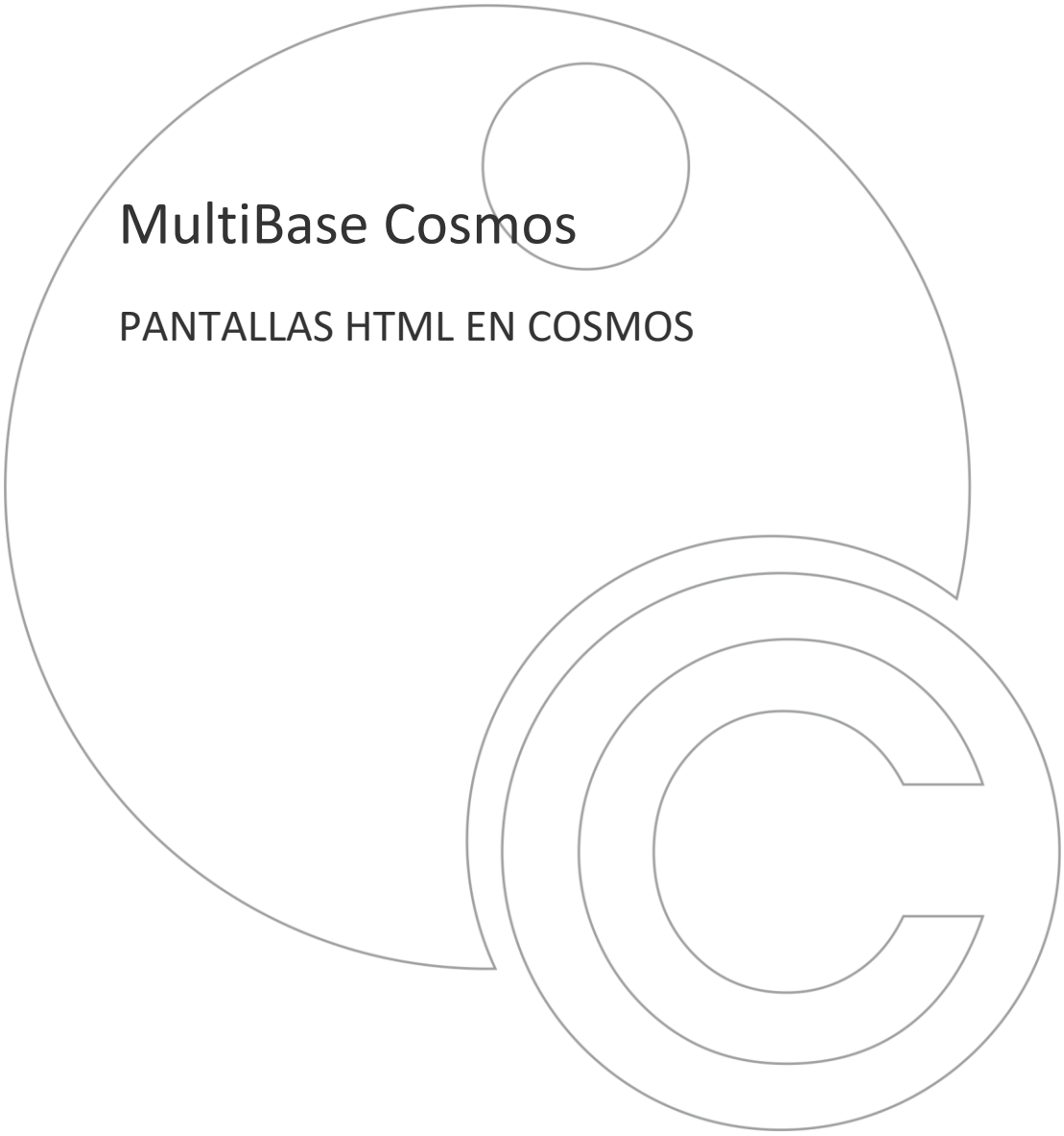
MultiBase Cosmos

PANTALLAS HTML EN COSMOS



BASE100

BASE 100, S.A.
www.base100.com

Índice

1 Formularios HTML en Cosmos

A partir de la versión 8.6, Cosmos incorpora la posibilidad de utilizar formularios diseñados directamente en HTML, permitiendo mayor flexibilidad visual, integración con estilos personalizados y capacidades adicionales mediante JavaScript.

1.1 Estructura del proyecto

Para habilitar formularios HTML en Cosmos, deben cumplirse las siguientes condiciones dentro del proyecto:

1.2 Carpeta htmlforms

En la raíz del proyecto debe existir una carpeta llamada «htmlforms». Dentro de la carpeta «htmlforms» se almacenarán los ficheros HTML correspondientes a las clases de los formularios que se mostrarán en HTML.

Dentro de ella se debe incluir un directorio CSS, que contendrá:

- Las hojas de estilo propias de Cosmos (solamente si existen forms traducidos con form2html.exe).
- Los archivos CSS adicionales que desee incluir el usuario.

1.3 Correspondencia entre formularios Cosmos y HTML

Un formulario de Cosmos está definido como una clase derivada de la clase Form.

Si se desea que dicho formulario se muestre utilizando un HTML personalizado, en la carpeta «htmlforms» debe existir un archivo HTML cuyo nombre siga esta nomenclatura:

```
modulo##clase.html
```

Donde:

modulo	Nombre del módulo donde está definida la clase Form.
Clase	Nombre de la clase del formulario.

Ejemplo:

En el módulo «ventas.smd» se define la clase «pedido», que deriva de la clase Form. El formulario HTML correspondiente se debería llamar «ventas##pedido.html» y estar situado en la carpeta «htmlforms» del proyecto.

2 Activación del modo HTML

Para que Cosmos utilice formularios HTML, es obligatorio definir una variable de entorno:

HTMLFORMSACTIVE=TRUE

Cuando esta variable está activa:

- Si existe un archivo HTML correspondiente al formulario, Cosmos mostrará la versión HTML del formulario.
- Si no existe el archivo HTML, el formulario se mostrará en modo tradicional.

Es totalmente posible mezclar formularios HTML con formularios nativos de Cosmos dentro de una misma aplicación.

3 Correspondencia de controles entre Cosmos tradicional y HTML

Esta sección describe cómo se traducen los controles clásicos de Cosmos al entorno HTML. Cada tipo de control tendrá su representación en HTML o, en caso de no tener equivalencia directa, se explicará su comportamiento.

Es un requisito indispensable que cada control en un formulario Cosmos tenga nombre, y que el atributo «id» del control correspondiente dentro de la página HTML tenga como valor el nombre del control del formulario Cosmos.

3.1 Controles con equivalencia HTML directa

Control Box

Un control Box de Cosmos se corresponderá con un elemento «div» dentro de la página HTML. Así mismo, cada una de las páginas del control Box tendrá su correspondiente elemento «div» como hijo del elemento «div» del control Box.

Por ejemplo, para un control Box llamado «box_principal» con dos páginas, su correspondencia en HTML será la siguiente:

```
<div id="box_principal" class="box_principal_class">
    <div id="box_principal#01" name="box_principal" page="01"
style="display:block;width:100%;height:100%;">
    </div>
    <div id="box_principal#02" name="box_principal" page="02"
style="display:block;width:100%;height:100%;">
    </div>
</div>
```

Control Text

Un control Text de Cosmos se corresponderá con un elemento «span» dentro de la página HTML.

Por ejemplo, para un control Text llamado «texto_descripcion» con el texto «Descripción del producto», su correspondencia en HTML será la siguiente:

```
<span id="texto_descripcion" class="texto_descripcion_class cosmos-text"
tabindex="0">Descripción del producto</span>
```

Control Edit Field

Un control Edit Field de Cosmos, dependiendo de sus características, tendrá las siguientes correspondencias:

- Edit Field simple:

Se corresponderá con un elemento «input» dentro de la página HTML.

Por ejemplo, un control Edit Field llamado «txt_codigo»:

```
<input id="txt_codigo" placeholder="Código cliente" class="txt_codigo_class cosmos-
editfield"/>
```

- Edit Field multilínea:

Se corresponderá con un elemento «textarea» dentro de la página HTML.

Por ejemplo, un control edit field llamado «txt_codigo»:

```
<textarea id="txt_codigo" class="txt_codigo_class cosmos-editfield"> </textarea>
```

- Edit Field combo sin spin:

Se corresponderá con un elemento «div» con un «input» y un «button» dentro de la página HTML.

El atributo «id» del campo «input» será el del Edit Field, seguido de la cadena «#input».

El atributo «id» del campo «button» será el del Edit Field, seguido de la cadena «#combobutton#01».

Por ejemplo, un control Edit Field llamado «txt_provincia»:

```
<div id="txt_provincia" class="txt_provincia_class cosmos-editfield-combo">
    <input name="txt_provincia" type="text" id="txt_provincia#input" class="cosmos-input-
combo" placeholder="Código provincia"/>
    <button name="txt_provincia" id="txt_provincia#editcombobutton#01" class="cosmos-
editfield-button-combo">
        <svg width="16" height="16" viewBox="0 0 16 16" fill="black"
xmlns="http://www.w3.org/2000/svg">
            <polygon points="8,13 1,5 16,5"></polygon>
        </svg>
    </button>
</div>
```

- Edit Field combo con spin:

Se corresponderá con un elemento «div» con un «input» y dos «button» dentro de la página HTML.

El atributo «id» del campo «input» será el del Edit Field, seguido de la cadena «#input».

El atributo «id» de los campos «button» será el del Edit Field, seguido de la cadena «#combobutton#01» y «#combobutton#02» respectivamente.

Por ejemplo, un control Edit Field llamado «editFieldComboSpin»:

```
<div id="editFieldComboSpin" class="editFieldComboSpin_class cosmos-editfield-spin">
  <input name="editFieldComboSpin" type="text" id="editFieldComboSpin#input" class="cosmos-
input-combo"/>
  <div class="cosmos-editfield-button-group-spin">
        <button name="editFieldComboSpin" id="editFieldComboSpin#editcombobutton#01"
class="cosmos-editfield-button-combo-spin"/>
        <svg width="16" height="16" viewBox="0 0 16 16" fill="black"
xmlns="http://www.w3.org/2000/svg">
          <polygon points="8,4 0,8 16,8"></polygon>
        </svg>
        <button name="editFieldComboSpin" id="editFieldComboSpin#editcombobutton#02"
class="cosmos-editfield-button-combo-spin"/>
        <svg width="16" height="16" viewBox="0 0 16 16" fill="black"
xmlns="http://www.w3.org/2000/svg">
          <polygon points="8,5 0,1 16,1"></polygon>
        </svg>
  </div>
</div>
```

Control Drop List

Un control Drop List de Cosmos se corresponderá con un elemento «div» que tendrá dos hijos «div»: uno para mostrar el texto del elemento seleccionado, y el otro para mostrar la lista de selección de elementos.

Por ejemplo, un control Drop List llamado «ctr_droplst»:

```
<div id="ctr_droplst" class="ctr_droplst_class cosmos-droplist-multicolumn" tabindex="0">
    <div id="ctr_droplst_selector" class="cosmos-droplist-multicolumn-selected">
        <span class="cosmos-droplist-multicolumn-selected-text"> </span>
        <span class="cosmos-droplist-multicolumn-arrow">&lt;</span>
    </div>
    <div class="cosmos-droplist-multicolumn-options-container" id="ctr_droplst_container">
</div>
</div>
```

Control Drop Edit

Un control Drop Edit de Cosmos se corresponderá con un elemento «div» que tendrá dos hijos: un elemento «input» para el campo de edición, y un «div» para el botón.

El atributo «id» del campo «input» será el mismo que el del control Drop Edit, seguido de «#edit».

El atributo «id» del campo «list» será el mismo que el del control Drop Edit, seguido de «#list».

Por ejemplo, un control drop edit llamado «ctr_dropEdit»:

```
<div id="ctr_dropEdit" class="ctr_dropEdit_class cosmos-dropedit-multicolumn" tabindex="0">
  <input id="ctr_dropEdit#edit" name="ctr_dropEdit#edit" list="ctr_dropEdit#list" type="text"
style="display:block;width:100%;height:100%;border:0px;padding:0px" tabindex="-1"
autocomplete="off" data-items="[]" data-colnames="ctrdropedit"/>
  <svg class="icon-arrow" xmlns="http://www.w3.org/2000/svg" viewBox="0 0 24 24"> <polyline
points="6 9 12 15 18 9"> </polyline></svg>
  <div class="cosmos-dropedit-multicolumn-dropdown" id="ctr_dropEdit#list" style="width:auto"
tabindex="-1">
    <div class="cosmos-dropedit-header"> </div>
       <div class="cosmos-dropedit-list"> </div>
  </div>
</div>
```

Control Push Button

Un control Push Button de Cosmos se corresponderá con un elemento «button» dentro de la página HTML.

Por ejemplo, un control Push Button llamado «ctr_boton»:

```
<button id="ctr_boton" class="ctr_boton_class cosmos-button" tabindex="0">Texto del
botón</button>
```

Control Radio Button

Un control Radio Button de Cosmos se corresponderá con un elemento «fieldset» que tendrá un hijo «input» de tipo «radio» y un hijo «label» por cada elemento del Radio Button.

El atributo «id» de cada control «input» será el mismo que el del control Radio Button, seguido del carácter «#» y del ordinal del elemento dentro del Radio Button.

Por ejemplo, un control Radio Button llamado «rb_propiedades" con cuatro opciones:

```
<fieldset id="rb_propiedades" class="rb_propiedades_class cosmos-radio-button" tabindex="0">
  <input type="radio" id="rb_propiedades#1" value="AllowShiftColumns" name="rb_propiedades"/>
  <label for="rb_propiedades#1">AllowShiftColumns</label>
  <br/>
  <input type="radio" id="rb_propiedades#2" value="Background" name="rb_propiedades"/>
  <label for="rb_propiedades#2">Background</label>
  <br/>
  <input type="radio" id="rb_propiedades#3" value="Bold" name="rb_propiedades"/>
  <label for="rb_propiedades#3">Bold</label>
  <br/>
  <input type="radio" id="rb_propiedades#4" value="Comment" name="rb_propiedades"/>
  <label for="rb_propiedades#4">Comment</label>
  <br/>
  <input type="radio" id="rb_propiedades#5" value="Disabled" name="rb_propiedades"/>
  <label for="rb_propiedades#5">Disabled</label>
  <br/>
</fieldset>
```

Control Check Box

Un control Check Box de Cosmos se corresponderá con un elemento «fieldset» que tendrá un hijo «input» de tipo «check» y un hijo «label».

El atributo «id» del control «input» será el mismo que el del control Check Box, seguido del texto «#input».

Por ejemplo, un control Check Box llamado «check_bold»:

```
<fieldset id="check_bold" class="check_bold_class cosmos-check" tabindex="0">
  <input type="checkbox" name="check_bold" id="check_bold#input" tabindex="-1"/>
  <label for="check_bold">Bold</label>
</fieldset>
```

Control Percentage Box

Un control Percentage Box de Cosmos se corresponderá con un elemento «progress».

Por ejemplo, un control Percentage Box llamado «ctr_Porcentaje»:

```
<progress id="ctr_Porcentaje" name="ctr_Porcentaje" value="0" max="100"
class="ctr_Porcentaje_class" tabindex="0"> </progress>
<input type="number"> con formato/validación CSS+JS
```

Control Button Group

Un control Button Group de Cosmos se corresponderá con un elemento «div» con controles hijos de tipo «button» dentro de la página HTML.

El «id» de cada botón hijo será el mismo que el del control Button Group, seguido de un carácter almohadilla «#» y un número que se corresponderá con el ordinal del botón dentro del grupo.

Por ejemplo, un control Push Button llamado «id_buttonGroup»:

```
<div id="id_buttonGroup" class="id_buttonGroup_class">
    <button id="id_buttonGroup#1" name="id_buttonGroup">1</button><br/>
    <button id="id_buttonGroup#2" name="id_buttonGroup">2</button><br/>
    <button id="id_buttonGroup#3" name="id_buttonGroup">3</button><br/>
    <button id="id_buttonGroup#4" name="id_buttonGroup">4</button>
</div>
```

Control Box Group

Un control Box Group de Cosmos se corresponderá con un elemento «div» con controles hijos de tipo «form» dentro de la página HTML.

El «id» de cada box hijo será el mismo que el del control Box Group, seguido de un carácter almohadilla «#» y un número que se corresponderá con el ordinal del control dentro del grupo.

Por ejemplo, control Box Group llamado «id_boxGroup» con cuatro Box hijos:

```
<div id="id_boxGroup" class="id_boxGroup_class">
    <form id="id_boxGroup#1" name="id_boxGroup">Box 1</form>
    <form id="id_boxGroup#2" name="id_boxGroup">Box 2</form>
    <form id="id_boxGroup#3" name="id_boxGroup">Box 3</form>
    <form id="id_boxGroup#4" name="id_boxGroup">Box 4</form>
</div>
```

Control List Box

Un control List Box de Cosmos, dependiendo de sus características, tendrá las siguientes correspondencias:

- Lista árbol.

Un control List Box de tipo árbol de Cosmos se corresponderá con un elemento «div» dentro de la página HTML. Cada elemento hijo de la lista árbol será una combinación de elementos «ul» y «li» de HTML.

- Lista fichero.

Un control List Box de tipo fichero de Cosmos se corresponderá con un elemento «div» dentro de la página HTML. Este elemento «div» será un elemento contenedor, cuyo «id» será la cadena «container-cosmos-table#» seguido del nombre del control lista, que a su vez contendrá un elemento «table" con todas las filas de la lista.

Por ejemplo, control List Box de tipo «file» llamado «listboxFile»:

```
<div id="container-cosmos-table#listboxFile" name="listboxFile" class="listboxFile_class
cosmos-listbox-container" tabindex="0">
    <table id="listboxFile" class="cosmos-listbox">
        <thead>
        </thead>
        <tbody>
        </tbody>
    </table>
</div>
```

- Lista Column List

Un control List Box de tipo Column List de Cosmos se corresponderá con un elemento «div» dentro de la página HTML. Este elemento «div» será un elemento contenedor, cuyo «id» será la cadena «container-cosmos-table#» seguido del nombre del control lista, que a su vez contendrá un elemento «table» con todas las filas de la lista.

Por ejemplo, control List Box de tipo «Column List» llamado «Lst1»:

```
<div id="container-cosmos-table#Lst1" name="Lst1" class="Lst1_class cosmos-listbox-container"
tabindex="0">
    <table id="Lst1" class="cosmos-listbox">
        <thead>
            <tr>
                <th id="Lst1#headercolumn#0" name="Lst1">Codigo</th>
                <th id="Lst1#headercolumn#1" name="Lst1">Name</th>
            </tr>
        </thead>
        <tbody>
        </tbody>
    </table>
</div>
```

- Lista Normal

Un control List Box de tipo Normal de Cosmos se corresponderá con un elemento «div» dentro de la página HTML. Este elemento «div» será un elemento contenedor, cuyo id será la cadena «container-cosmos-table#» seguido del nombre del control lista, que a su vez contendrá un elemento «div» por cada elemento de la lista.

Por ejemplo, control List Box de tipo «Normal» llamado «Lst1»:

```
<div id="container-cosmos-table#Lst1" name="Lst1" class="Lst1_class cosmos-listbox-container"
tabindex="0">
    <div id="Lst1" class="cosmos-listbox-bycolumns">
    </div>
</div>
```

- Lista By Columns

Un control List Box de tipo By Columns de Cosmos se corresponderá con un elemento «div» dentro de la página HTML. Este elemento «div» será un elemento contenedor, cuyo «id» será la cadena «container-cosmos-table#» seguido del nombre del control lista, que a su vez contendrá un elemento «table» con todas las filas de la lista.

Por ejemplo, control List Box de tipo «By Columns» llamado «Lst1»:

```
<div id="container-cosmos-table#Lst1" name="Lst1" class="Lst1_class cosmos-listbox-container"
tabindex="0">
    <table id="Lst1" class="cosmos-listbox">
        <tbody>
        </tbody>
    </table>
</div>
```

Control Bitmap

Si el control Bitmap de Cosmos no tiene hijos, se corresponderá con un elemento «img». Si el control Bitmap tiene hijos, se corresponderá con un control «div».

Por ejemplo, control bitmap sin hijos llamado «imagen_bmp»:

```
<img id="imagen_bmp" class="imagen_bmp_class" src=”bitmap.bmp”> </img>
```

Bar Control

Un control Bar de Cosmos se corresponderá con un elemento «div» dentro de la página HTML. Así mismo, cada una de las páginas del control Bar tendrá su correspondiente elemento «div» como hijo del elemento «div» del control Box.

Por ejemplo, para un control Box llamado «bar_ctr» con dos páginas, su correspondencia en HTML será la siguiente:

```
<div id="bar_ctr" class="bar_ctr_class cosmos-bar">
    <div id="bar_ctr#01" page="01" style="display:block;width:100%;height:100%;">
        <span> Texto </span>
    </div>
</div>
```

Control Panel

Un control Panel de Cosmos, dependiendo de sus características, tendrá las siguientes correspondencias:

- Si no es multilínea:

  Se corresponderá con un elemento «span» dentro de la página HTML. Por ejemplo, para un control Panel llamado «id_panel_operation»:

```
<span id="id_panel_operation" class="id_panel_operation_class"> </span>
```

- Si es multilínea:

  Se corresponderá con un elemento «span» con un elemento hijo «span» por cada línea.

Control Menu

Un control Menu de Cosmos se corresponderá con un elemento de tipo «fieldset». Cada opción de menú se corresponderá con un elemento del tipo «button».

El «id» de cada botón hijo será el mismo que el del control Menu, seguido de la cadena «#button» y un número que se corresponderá con el ordinal del control dentro del grupo.

Por ejemplo, para un control Menu de nombre «ctrMenu”:

```
<fieldset id="ctrMenu" class="ctrMenu_class" tabindex="0">
  <Button name="ctrMenu" id="ctrMenu#button#01">Opcion 1</Button>
  <Button name="ctrMenu" id="ctrMenu#button#02">Opcion 2</Button>
  <Button name="ctrMenu" id="ctrMenu#button#03">Opcion 3</Button>
</fieldset>
```

Control Slider

Un control Slider de Cosmos se corresponderá con un elemento de tipo «input range».

Por ejemplo, para un control Slider de nombre «mislider»:

```
<input id="mislider" name="mislider" type="range" min="0" max="10" class="mislider_class"/>
```

Tab Control

Un Tab Control de Cosmos se corresponderá con un elemento «div» de HTML que a su vez tendrá como hijos un elemento «button» y un elemento «div» por cada página.

El «id» de cada botón hijo será el mismo que el del control Tab, seguido de la cadena «#button» y un número que se corresponderá con el ordinal del control dentro del grupo.

El «id» de cada página hija («div») será el mismo que el del control Tab, seguido de la cadena «#» y un número que se corresponderá con el ordinal de la página dentro del Tab.

Por ejemplo, un Tab Control llamado «MiTab», con dos pestañas:

```
<div id="Mitab" class="Mitab_class">
    <div id="Mitab#01" page="01" name="Mitab" class="Mitab_page_01_class"
style="display:block;width:100%;height:100%;">
        <span> </span>
    </div>
    <div id="Mitab#02" page="02" name="Mitab" class="Mitab_page_02_class"
style="display:none;width:100%;height:100%;">
        <span> </span>
    </div>
    <button id="Mitab#button#01" name="Mitab" class="Mitab_button_01_class">Page 1</button>
    <button id="Mitab#button#02" name="Mitab" class="Mitab_button_02_class">Page2</button>
</div>
```

3.2 Controles sin equivalencia HTML nativa o limitada

Los siguientes controles no tiene equivalencia en HTML:

- Control Grid no tienen correspondencia HTML. Cuando se utiliza un formulario HTML, el control Grid se muestra como en un formulario tradicional.
- Control Split (Horizontal, Vertical, Split).
- Control List Box en árbol Muilticolumna.
- Control UserControl.
- ActiveX.

4 Ejecución de funciones Javascript desde Cosmos

Cosmos permite invocar funciones JavaScript definidas en el HTML del formulario.

El mecanismo general es:

- Definir la función JavaScript en el HTML o en un archivo JS local que esté enlazado en el fichero HTML.
- Desde la aplicación Cosmos, llamar la función mediante el método **RunJavascriptFunction**.

Definición de la función:

```
RunJavascriptFunction(functionStr as Char, parameterStr as Char)
```

Parámetros:

functionStr.	Cadena de caracteres con el nombre de la función.
parameterStr.	Cadena de caracteres con los parámetros de la función.

Retorno:

La función no retorna ningún valor.

Ejemplo:

```
// En el fichero HTML
function actualizarTotales(str) {
    // Guardamos en "obj" el parámetro pasado en formato JSON
    obj = JSON.parse(str);
    // lógica de actualización
}
```

```
// Desde Cosmos
form.RunJavascriptFunction ("actualizarTotales", "parametros");
```

El sistema se encarga de ejecutar la función en el contexto del formulario HTML cargado.

5 Variables de entorno

HTMLFORMSACTIVE.

Si se define con valor TRUE, activa el modo HTML. Si no se define o se define con valor distinto a TRUE, todos los formularios Cosmos se mostrarán en modo tradicional.

HTMLFORMSDISABLECONTEXTMENU.

Por defecto, al pulsar con el botón derecho del ratón sobre una pantalla Cosmos en HTML, se muestra un menú propio de Chromium (ver sección Motor de renderizado). Este menú es útil para depurar el aspecto visual de las aplicaciones. Se recomienda que, en un entorno de producción, se defina la variable HTMLFORMSDISABLECONTEXTMENU con valor TRUE.

6 Traductor de formularios Cosmos a HTML

Cosmos dispone una aplicación llamada «form2html.exe» que permite traducir automáticamente los formularios de un módulo o de una aplicación Cosmos a HTML.

Línea de comando:

```
form2html.exe -prj <proyecto> [-all| -smd <módulo>] [-help]
```

Parámetros:

-prj <proyecto>	Indica el proyecto del cual se desean traducir clases Form a HTML
-all	Indica que se traducirán las clases Form de todos los módulos del proyecto.
-smd <módulo>	Indica que se traducirán las clases Form de un determinado módulo del proyecto.
-help	Muestra la información de los parámetros.

Tras la ejecución de «form2html.exe», los ficheros HTML correspondientes a las clases Form traducidas se almacenarán en la carpeta «htmlforms».

Los ficheros HTML generados por form2html necesitarán los archivos «CSS» (ver anexo **archivos css**) incluidos en la instalación de Cosmos, que deberán incluirse en la carpeta «CSS» dentro de «htmlforms».

Estos ficheros HTML mostrarán una representación fiel de todos los controles incluidos en la clase Form correspondiente, tanto en posición en pantalla, dimensiones, Font, colores de fondo y texto, etc.

El programa form2html.exe es una ayuda para realizar una traducción rápida y fiel de las clases Form incluidas en un módulo o en un proyecto Cosmos.

Cosmos permite la ejecución de aplicaciones con Forms HTML sin necesidad de generar los ficheros HTML con form2html.exe. El desarrollador podrá crear sus propias páginas HTML utilizando herramientas externas, siempre y cuando cumpla las reglas de correspondencia de controles Cosmos-HTML descritas en secciones anteriores.

7 Limitaciones

A continuación, se enumeran las características no soportadas por las pantallas HTML y los controles no se traducen.

Las siguientes características no están soportadas:

- No es posible importar JavaScript remoto. Solo se permite utilizar scripts locales incluidos en el proyecto.
- COSMOSVISUALMODE (CUSTOMCOLORS, CUSTOMCONTROLS, CUSTOMFONTS).
- SKIN en listas.
- Funciones de estilos/personalización en controles LIST BOX.
- El nombre de los controles no debe tener caracteres especiales, vocales con tilde o letra «ñ».
- Las máscaras.

Los siguientes controles no tiene equivalencia en HTML:

- Control Grid no tienen correspondencia HTML.
  Cuando se utiliza un formulario HTML, el control Grid se muestra como en un formulario tradicional.
- Control Split (Horizontal, Vertical, Split) no se traducen.
- Control List Box en árbol Muilticolumna.
- Referencia a los iconos de los controles Push Button.

8 Motor de renderizado

Para renderizar los formularios en formato HTML, Cosmos utiliza WebView2, es un control de desarrollo de Microsoft que permite incrustar contenido web (HTML, CSS, JavaScript) directamente en aplicaciones nativas de Windows. Este control utiliza el motor de representación de Microsoft Edge (basado en Chromium) para mostrar dicho contenido.

Este control está incluido en versiones Windows 10 y posteriores, por lo que para el correcto funcionamiento de las pantallas HTML en Cosmos no será necesario realizar ningún tipo de instalación de software de terceros.

Archivos css

En el proyecto «samples» de Cosmos se distribuyen los archivos «css»

Fichero	Contenido
cosmos.css	Características generales para todos los controles. Importa los archivos css de todos los controles. El pseudo elemento ::placeholder. Clases: «.error» «.noerror» «.Form».
styles.css	Variable css incluidas en la pseudo-clase «:root» donde se definen algunos colores.
bar.css	Define características específicas para el control Bar.
button.css	Define características específicas para el control Button.
check.css	Define características específicas para el control Check.
dropedit-multicolumn.css	Define características específicas para el control Drop Edit Multicolumna.
droplist-multicolumn.css	Define características específicas para el control Drop List Multicolumna.
editfield.css	Define características específicas para el control Edit Field, Edit Field combo y Edit Field spin.
listbox.css	Define características específicas para el control List Box: «normal» «por columnas» y «multicolumnas»
listtree.css	Define características específicas para el control List Box árbol.
percentagebox.css	Define características específicas para el control PorcentajeBox.
radiobutton.css	Define características específicas para el control Radio Button.
slider.css	Define características específicas para el control Slider.
Spin.css	Define características específicas para el control Spin.
text.css	Define características específicas para el control Text.