MultiBase Cosmos

Notas a la versión 5.6

BASE100

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

Índice

[NTCO56v1.2]

1. Incompatibilidades

En esta versión de Cosmos es obligatorio recompilar las aplicaciones compiladas con versiones anteriores.

Así mismo, es necesario modificar los programas de la aplicación en caso de utilizar la prnpag32.dll. En esta dll se ha cambiado el tipo (de smallint a integer) en algunos de los parámetros de sus funciones.

Las lista de funciones que se han modificado son las siguientes:

rootControl(HPage as integer) return **integer**
nextControl(HPage as integer, idm as **integer**) return **integer**
controlByName(HPage as integer, usrid as char) return **integer**
controlByUsrIdm(HPage as integer, usridm as **integer**) return **integer**
getControlName(HPage as integer, idm as **integer**) return char
getControlUsrIdm(HPage as integer, idm as **integer**) return **integer**
childControl(HPage as integer, idm as **integer**) return **integer**
getWidth(HPage as integer, idm as **integer**) return integer
getHeight(HPage as integer, idm as **integer**) return integer
parentControl(HPage as integer, idm as **integer**) return **integer**
setControlText(HPage as integer, idm as **integer**, text as char)
count(HPage as integer, idm as **integer**) return integer
addBand(HPage as integer, idmGroup as **integer**, idmBand as **integer**) return boolean
remainingSpace(HPage as integer, idm as **integer**) return integer
setPropStr(HPage as integer, idm as **integer**, prop as char, text as char)
setPropInt(HPage as integer, idm as **integer**, prop as char, valueprop as integer)
getPropStr(HPage as integer, idm as **integer**, prop as char, var text as char, len as smallint)
getPropInt(HPage as integer, idm as **integer**, prop as char) return integer
moveControl(HPage as integer, idm as **integer**, x as smallint, y as smallint) return boolean
changeSizeControl(HPage as integer, idm as **integer**, xSize as smallint, ySize smallint) return boolean
getXPos(HPage as integer, idm as **integer**) return integer
getYPos(HPage as integer, idm as **integer**) return integer
setXPos(HPage as integer, idm as **integer**, xPos as smallint)
setYPos(HPage as integer, idm as **integer**, yPos as smallint)
setBoxRoundCornerRadio(HPage as integer, idm as **integer**, radio as smallint)

2. Implementaciones

2.1 Cosmos

2.1.1 Runtime de Cosmos

- Nueva dll coshttpdll. Esta dll permite establecer conexiones con servidores web a través del protocolo http.
- Multi-idioma. Se podrá cambiar en ejecución el idioma en el que se muestras las etiquetas (label) y los comentarios (comments) de los controles de una aplicación desarrollada en Cosmos (**ver apartado 4**).
- Posibilidad de realizar una llamada a una función de Java desde un programa Cosmos (**ver apartado 5**).
- Evento ListSpreadSheetColChange. Este evento se lanza cuando en un control List Box (string o sql), estando habilitada la navegación por sus celdas (método SetListSpreadSheetNavigation), se produce un cambio de celda en la misma fila.
- Métodos en la clase SimpleControl que permiten:
  - Aceptar y cancelar la edición en los controles List Box (listas editables): ListInvokeAcceptEdit y ListInvokeCancelEdit (**ver apartado 6.1**).
  - Guardar y retornar los atributos de la lista (control List Box de tipo string y sql) en tiempo de ejecución: métodos SetListStatusStr y GetListStatusStr (**ver apartado 6.1**).
  - Retornar una cadena de caracteres con la configuración de una lista agrupada (**ver apartado 6.1**).
- Método de la clase Char que permite reemplazar caracteres en un objeto Char.
- Funciones en la dll TTXMLDLL que permiten generar un nuevo documento a partir de un fichero xml y una hoja de estilo. También se podrá modificar el contenido y/o nombre de un nodo de un documento xml.

2.1.2 Entorno de desarrollo

- La información de la pestaña Find in Files de la ventana Output se mostrará encolumnada para facilitar la búsqueda.
- Se ha añadido una opción más al menú popup que permite copiar una línea de la lista al portapapeles (clipboard).
- Code insight.
  1. Posibilidad de cambiar el color de los tokens en el editor de código.

     En la pestaña *Editor* de la opción ***Settings*** del menú **Tools** del entorno de desarrollo se podrá cambiar el color de: keyword (palabras reservadas, color por defecto: rojo), identificadores (p.e nombre de los controles, color por defecto: negro), números (color por defecto: azul), comentarios (color por defecto: verde), delimitadores (color por defecto: negro) y cadena de caracteres (color por defecto: azul).

     La información se almacenará automáticamente en el fichero de configuración de Cosmos.
  2. Al pulsar la combinación de teclas [Ctrl]+[Espacio] en la sección Code de cualquiera de las clases definidas en un módulo de Cosmos se mostrará una lista con los objetos, constantes, variables, controles, métodos y propiedades que comienzan por la(s) letra(s) indicada(s).

3. Posibilidad de localizar la definición de un ítem de la aplicación de una manera más sencilla. En esta nueva versión, al pulsar [Ctrl]+[t] sobre el ítem, el cursor del ratón se posicionará en el lugar donde éste esté definido.

   Limitaciones en los puntos 2 y 3:

   I. No incluye los comandos.
   II. Si el control es un tab control se posicionará en el control, no en la página del tab.

2.2 Gestor de base de datos (CTSQL)

A partir de esta versión el gestor de base de datos generará opcionalmente un fichero de estadísticas donde se mostrará información de la ejecución de instrucciones SQL en la conexión a la base de datos:

- Prepare, Open, Execute, Fecth de cada instrucción.
- Tiempo empleado en la ejecución de cada una de estas funciones para cada instrucción.

Si el cliente del motor es Cosmos, se contabilizarán las instrucciones que el runtime envía al motor en los comandos predefinidos (EditUptate, Add, …) y las instrucciones SQL puras ejecutadas con los métodos de la clase SqlCursor, SqlStatement y SqlServer.

Para activar/desactivar las estadísticas se han implementado los siguientes mecanismos:

1. Mediante una instrucción SQL:

```
set statistics to 1 (activa las estadísticas)
set statistics to 0 (desactiva las estadísticas)
```

   El fichero se genera cuando se ejecuta la instrucción “set statistics to 0” o cuando el cliente se desconecta del servidor.

2. Mediante la variable de entorno CTSQLSTATISTICS:

   Los posibles valores son: TRUE/FALSE o YES/NO.

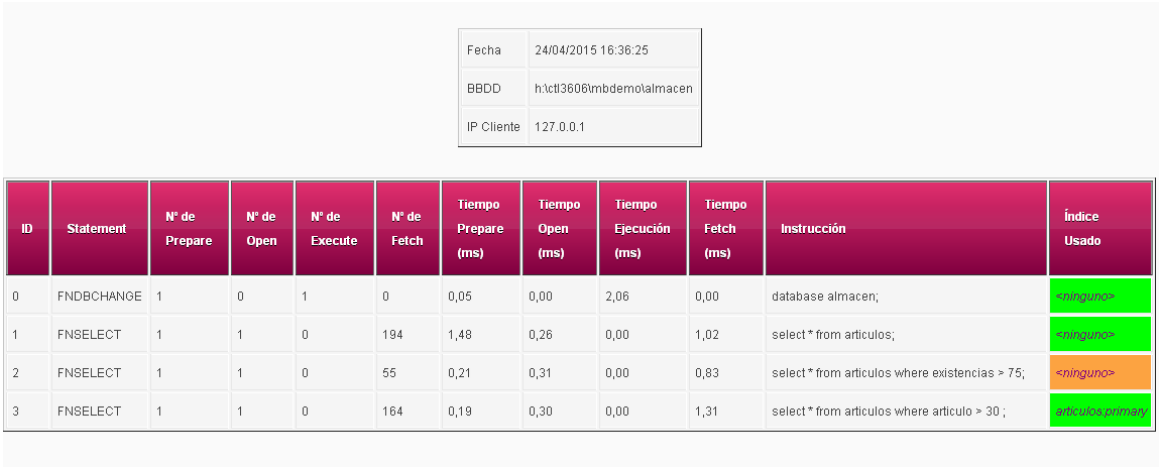
   TRUE activa las estadísticas y FALSE las desactiva.

   Se puede definir en el fichero de configuración del motor (ctsql.ini) o en el Environment de la conexión de Cosmos o con el método PutEnv de la clase Module o con el método SetValue de la clase SqlServer. La llamada al método se deberá realizar antes de la conexión a la base de datos.

   El fichero de estadísticas se genera cuando finaliza la conexión a la base de datos.

En el caso de que se utilice la variable de entorno y la instrucción SQL, el fichero se generará cuando se ejecute la sentencia SQL (set statistics to 0).

La salida del fichero es la siguiente:



Fecha	24/04/2015 16:36:25
BBDD	h:\ctl3606\mbdemo\almacen
IP Cliente	127.0.0.1

ID	Statement	Nº de Prepare	Nº de Open	Nº de Execute	Nº de Fetch	Tiempo Prepare (ms)	Tiempo Open (ms)	Tiempo Ejecución (ms)	Tiempo Fetch (ms)	Instrucción	Índice Usado
0	FNDBCHANGE	1	0	1	0	0,05	0,00	2,06	0,00	database almacen;	<ninguno>
1	FNSELECT	1	1	0	194	1,48	0,26	0,00	1,02	select * from articulos;	<ninguno>
2	FNSELECT	1	1	0	55	0,21	0,31	0,00	0,83	select * from articulos where existencias > 75;	<ninguno>
3	FNSELECT	1	1	0	164	0,19	0,30	0,00	1,31	select * from articulos where articulo > 30 ;	articulos.primary

Por cada frase SQL, los datos que muestra son:

ID	Identificador interno de la instrucción.
Statement	Código interno de la instrucción SQL.
Nº de Prepare:	Indica el número de veces que se ha preparado la frase SQL.
Nº de Open	Indica el número de veces que se ha abierto el cursor de la frase SQL, en el caso de que se trate de una query.
Nº de Execute	Indica el número veces que se ha ejecutado la frase SQL.
Nº de Fetch	Indica el número veces que se ha ejecutado la instrucción Fecth sobre la frase SQL en el caso de que se trate de una query.
Tiempo Prepare (ms)	Tiempo (en milisegundos) que ha tardado el gestor en ejecutar las 'n' instrucciones Prepare.
Tiempo Open(ms)	Tiempo (en milisegundos) que ha tardado el gestor en ejecutar las 'n' instrucciones Open.
Tiempo Ejecución (ms)	Tiempo (en milisegundos) que ha tardado el gestor en ejecutar las 'n' instrucciones Execute.
Tiempo Fetch (ms)	Tiempo (en milisegundos) que ha tardado el gestor en ejecutar las 'n' instrucciones Fetch.
Instrucción	Frase SQL.
Índice usado	Nombre del índice seleccionado por el motor. Esta columna además se mostrará con un color de fondo. Estos colores serán los siguientes: • Verde. La celda se mostrará con este color cuando: La instrucción SQL precisa de un índice y el gestor ha seleccionado uno. Si la una instrucción no precisa índice para su optimización, p.e: "database stock".

- Naranja. La celda será de color naranja si la sentencia SQL ejecutada precisa de un índice para ser optimizada y no se ha utilizado ninguno. P.e “select * from artículos where existencias > 75”, y no existe un índice que tenga como primer campo “existencias”.

3. Mejoras

- GetUrlFileEx. Se ha modificado esta función para que admita los protocolos sftp, scp y https.
- Método CallWebService. Se ha ampliado el tamaño del parámetro headerSend de 10.240 a 32.767 caracteres.
- Se ha implementado un nuevo método, llamado CallWebServiceEx. Este método permite la conexión con un servicio web. La diferencia entre este método y el método CallWebService radica en que, como parámetro, recibe, en lugar de una cadena de caracteres con la información que se enviará al servidor, la ruta de un fichero que contendrá la información a enviar al servicio web. De esta manera se elimina la limitación de 32.767 caracteres en el parámetro del método CallWebService.
- Se ha eliminado la limitación de Cosmos que impedía depurar módulos con más de 32.767 líneas.
- Se ha eliminado la limitación de Cosmos que impedía editar módulos con más de 32.767 líneas.
- Se ha eliminado la limitación de Cosmos que provocaba que los listados con más de 32.767 elementos no se visualizaran en su totalidad en el Preview.
- Posibilidad de ejecutar solo la instrucción seleccionada en el Sql-Interactivo.
- Posibilidad de mostrar totales en las cabeceras de grupos en las listas agrupadas cuando se haya seleccionado como modo de presentación el modo compacto sin detalle y el nodo se encuentre cerrado. Para ello se ha implementado un nueva variable de entorno: SHOWAGGSINCOMPACTMODE.

4. Multi-idioma

A partir de esta versión se podrá cambiar en ejecución el idioma en el que se muestran las etiquetas (label) y los comentarios (comments) de los controles de una aplicación desarrollada en Cosmos. De esta manera, no será necesario disponer de una versión diferente de la aplicación para cada idioma.

Las etiquetas y los comentarios se almacenarán en un fichero junto con los textos traducidos.

La estructura del fichero será la siguiente:

```
Etiqueta=Etiqueta traducida
```

Ejemplo:

```
Control de Acceso=Access Control
Password:=Password:
Codigo Cliente=Customer Code
Empresa=Company
Salir=Exit
```

Si la etiqueta del control no coincide con ninguna de las indicadas en el fichero se mostrará la etiqueta original.

Se podrá indicar al runtime que usará el fichero de traducción de dos maneras:

1. Utilizando una variable de entorno MULTILANGUAGEFILE.

```
[Environment]
MULTILANGUAGEFILE=c:\cosmos\project\Almafac\ingles.txt
```

2. Con un nuevo parámetro del runtime de Cosmos.

-multilanguagefile	Indica la ruta absoluta relativa al proyecto del fichero de texto que contiene las etiquetas de los controles y su traducción.

Para que resulte más fácil detectar las etiquetas que no se han traducido, se ha implementado la posibilidad de que el runtime genere un fichero de log que muestre las etiquetas y los comentarios sin correspondencia en el fichero de traducción durante la ejecución de la aplicación. La ruta donde se creará el fichero se indicará a través de una variable de entorno, llamada MULTILANGUAGEDEBUGFILE, o mediante un nuevo parámetro del runtime: “-multilanguagedebugfile”.

No es obligatorio declarar la variable de entorno MULTILANGUAGEDEBUGFILE ni el parámetro, aunque sí es aconsejable durante la fase de traducción de la aplicación.

Los dos variables de entorno se declaran en la sección Environment del fichero INI de la aplicación.

Esta implementación solo afecta a las etiquetas de los controles asignadas en tiempo de diseño. Si se modifica la etiqueta de un control en tiempo de ejecución asignando un valor a la propiedad Label, esta etiqueta prevalecerá sobre la indicada en el fichero de traducción.

5. Llamada a un método o función Java desde Cosmos

A partir de esta versión de Cosmos se podrán realizar llamadas desde un módulo de Cosmos a un método o función de una clase desarrollada en Java.

Para ello es necesario:

1. Máquina virtual Java instalada.
2. Dll cosjavadll.dll.

   Es una dll intermedia que se encarga de la conexión entre Cosmos y Java (enlaza con la jvm.dll).
3. Dll que establece la conexión entre Cosmos y Java.

   Para establecer esta conexión se utiliza el mecanismo JNI.

   Es necesario que en la variable de entorno PATH del sistema operativo se indique la ruta donde se encuentran instaladas las librerías jvm.dll y msvcr100.dll de Java en la versión de 32 bits.
4. Declaración del método o función Java en el módulo de Cosmos.

   La sintaxis es:

   ```
   acceso javaclass package método_función_java(parámetros) [RETURN clase]
   ```

acceso	Este campo es obligatorio. Indica el tipo de acceso de la función: public, private o protected.
javaclass	Palabra reservada. Indica que la función que se está declarando es de un programa java.
package	Literal entrecomillado que indica el paquete y la clase donde está definido el método.
Método_funcion_java	Nombre de la función o método Java que se desea utilizar. Debe ir entre comillas.
parámetros	Lista de identificadores válidos en COOL, separadas por comas. Para designar los parámetros del método Java que tienen que mapear objetos de clase se utilizará la palabra reservada JavaObject.
RETURN	Campo opcional. Indica que la función devuelve un valor.
clase	Tipo de objeto que devuelve la función o método java.
5. Fichero java.options

   En este fichero se indicarán los parámetros necesarios para la configuración de la máquina virtual Java.

   ```
   -Djava.class.path=c:\samples\chart\java\ejemplo1
   -Xms8m
   -Xmx24m
   ```

   Si no existe este fichero en el directorio del proyecto el runtime de Cosmos buscará un directorio jars, dentro del proyectos de Cosmos.

El directorio jars debe incluir todos los jars necesarios para la ejecución de las clases Java.

La máquina virtual Java que se debe instalar ha de ser de 32bits.

5.1 Correspondencia entre tipos de datos Cosmos y Java en la definición de los parámetros de la función

Cosmos puede ejecutar métodos o funciones de clases Java en los que el tipo de dato de los parámetros sean: String, short, Short, double, Double, int, Integer, boolean y Boolean.

El tipo de dato de retorno debe ser un tipo de dato básico o un objeto de la clase String.

La correspondencia entre los tipos de dato de los parámetros en una función o método Java y una función Cosmos son los siguientes:

Java	Cosmos
String	char
short	smallint
Short	javaobject smallint
double	decimal
Double	javaobject decimal
int	integer
Integer	javaobject integer
boolean	boolean
Boolean	javaobject boolean

Si el parámetro de la función o método Java es un objeto de una clase básica en lugar de un tipo de dato básico, en la definición de la función en Cosmos se debe indicar la palabra JavaObject.

6. Nuevos métodos

6.1 Métodos de la Clase SimpleControl

- ListInvokeAcceptEdit. Este método permite aceptar la edición en los controles List Box. De este manera el programador podrá definir una tecla de función adicional que permita realizar la acción. La tecla [Enter] seguirá realizando la misma acción. La definición de un nuevo acelerador no inhabilita la tecla por defecto.

  Sintaxis:

  ```
  ListInvokeAcceptEdit()
  ```

- ListInvokeCancelEdit. Este método permite cancelar la edición en los controles List Box. De este manera el programador podrá definir una tecla de función adicional que permita realizar la acción. La tecla [Esc] seguirá realizando la misma acción. La definición de un nuevo acelerador no inhabilita la tecla por defecto.

  Sintaxis:

  ```
  ListInvokeCancelEdit()
  ```

- GetListMultiColumnGroupStr. Este método retorna el valor de la cadena de configuración de una lista agrupada en árbol creada con el método ShowListAsMultiColumnGroup o ShowMultiColumnGroupDlg.

  Sintaxis:

  ```
  GetListMultiColumnGroupStr() return Char
  ```

  Retorna:

  Una cadena de caracteres con el mismo formato que el pasado como primer parámetro en el método ShowListAsMultiColumnGroup.

- GetListStatusStr. Este método retorna una cadena de caracteres con atributos de las columnas de un control list box (tipo string o sql), drop edit y drop list.

  Los atributos que retorna por cada columna son:

COLNUMBER	Identificador de la columna.
COLWIDTH	Ancho de columna en píxeles.
COLVISIBLE	Estado de la columna (visible/no visible).
COLWIDTHCHARS	Ancho de la columna en número de caracteres.
COLPOSITION	Posición de la columna en la lista.

  Sintaxis:

  ```
  GetListStatusStr() return char
  ```

  Retorna:

  Cadena de caracteres.

- SetListStatusStr. Permite modificar los atributos de las columnas de un control List Box (string o sql), drop edit y drop list de tipo column list.

  Recibe como parámetro una cadena de caracteres con los atributos y su valor.

Sintaxis:

```
SetListStatusStr(statusStr as Char)
```

Parámetros:

statusStr	Atributos y su valor para las columnas de la lista. Formato : El separador de cada columna será el punto y coma (';') y los atributos de cada columnas irán separados por el signo de dos puntos (':').

Ejemplo:

"COLNUM-
BER=1:COLWIDTH=57:COLVISIBLE=1:COLWIDTHCHARS=11:COLPOSITION=1;COLNUMBER=2:COLWIDTH=57
:COLVISIBLE=1:COLWIDTHCHARS=11:COLPOSITION=2;COLNUMBER=3:COLWIDTH=57:COLVISIBLE=1:COLWI
DTHCHARS=11:COLPOSITION=3;COLNUMBER=4:COLWIDTH=57:COLVISIBLE=0:COLWIDTHCHARS=11:COLP
OSI-
TION=4;COLNUMBER=5:COLWIDTH=57:COLVISIBLE=1:COLWIDTHCHARS=11:COLPOSITION=5;COLNUMBER
=6:COLWIDTH=57:COLVISIBLE=1:COLWIDTHCHARS=11:COLPOSITION=6;COLNUMBER=7:COLWIDTH=57:CO
LVISI-
BLE=1:COLWIDTHCHARS=11:COLPOSITION=7;COLNUMBER=8:COLWIDTH=57:COLVISIBLE=1:COLWIDTHCHA
RS=11:COLPOSITION=8;"

6.2 Métodos de la Clase Module

- GetMemoryStatus. Este método permite consultar el estado de las memorias física y virtual del sistema operativo.

  Sintaxis:

  ```
  GetMemoryStatus(infoType as Integer, var memoryIfo as Decimal)
  ```

infoType	Este parámetro indica el estado de memoria que se desea consultar. Los valores posibles son: 1. Un número entre 0 y 100 que especifica el porcentaje aproximado de la memoria física que está en uso (0 indica que no hay uso de memoria y 100 indica el uso de memoria llena). 2. La cantidad de memoria física real, en bytes. 3. La cantidad de memoria física disponible en la actualidad, en bytes. 4. Número total de bytes que pueden ser almacenados en el fichero de paginación. 5. Número de bytes disponibles en el fichero de paginación. 6. Número total de bytes que pueden ser descritos en la porción del modo usuario del espacio de direcciones virtual del proceso que ejecuta la función. 7. Número de bytes de memoria no reservada ni asignada en la porción del modo usuario del espacio de direcciones virtual del proceso que ejecuta la función.

memoryInfo	Parámetro por referencia en el que se almacena el valor de la memoria que se desea consultar.

Retorna:

TRUE si ha podido ejecutar la función sin errores y False si el primer parámetro no está entre 1 y 7.

6.3 Métodos de la Clase Char

- Translate. Este método permite reemplazar caracteres en un objeto Char. Se reemplazarán los caracteres indicados en el primer parámetro por los que se indiquen en el segundo. El método modifica el valor del objeto Char.

  Sintaxis:

  ```
  Translate(stringToReplace as Char, replacementString as Char)
  ```

stringToReplace	Caracteres que se van a reemplazar en el objeto Char.
replacementString	Caracteres por los que se va a reemplazar.

7. Eventos

- ListSpreadSheetColChange. Este evento se lanza cuando en un control List Box(string o sql), estando habilitada la navegación por sus celdas (método SetListSpreadSheetNavigation), se produce un cambio de celda en la misma fila.

8. Nuevas funciones de las Apis

8.1 Funciones de la API TTXMLDLL

- TTXmlApplyXSLTAndSave. Esta función permite generar un documento nuevo a partir de un fichero xml y una hoja de estilo. El documento generado no tiene por qué ser un xml.

  Sintaxis:

  ```
  TTXmlApplyXSLTAndSave(xmlFileName as char, xsltFileName as char, outXmlFile as char, var bytes as integer) return integer
  ```

  Parámetros:

xmlFileName	Documento xml sobre el que se va a aplicar la transformación.
xsltFileName	Hoja de estilo.
outXmlFile	Documento generado.
bytes	Número de bytes que se han escrito en el fichero de salida.

  Retorna:

  Código de Retorna:

  0: Si no hay error.

  -1: Error al aplicar la hoja de estilo al documento.

  -2: Error al cargar o parsear la hoja de estilo.

  -3: Error al cargar o parsear el fichero XML.

  -4: Error al crear el contexto de transformación.

- TTXmlUpdateNodeContent. Esta función permite modificar el contenido de un nodo del documento XML.

  Sintaxis:

  ```
  TTXmlUpdateNodeContent(Doc as integer, node as integer, Content as Char, charset as integer)
  ```

  Parámetros:

doc	Identificador del documento XML al que pertenece el nodo que se desea cambiar
node	Identificador del nodo que se desea cambiar.
content	Texto que indica el nuevo contenido del nodo. Si es NULL, la función eliminará el contenido del nodo.
charset	Set de caracteres en el que están codificados los parámetros "content".

- TTXmlUpdateNodeName. Esta función permite modificar el nombre de un nodo del documento XML.

  Sintaxis:

  ```
  TTXmlUpdateNodeName(Doc as integer, node as integer, name as Char, charset as integer)
  ```

Parámetros:

doc	Identificador del documento XML al que pertenece el nodo que se desea cambiar
node	Identificador del nodo que se desea cambiar.
name	Texto que indica el nuevo nombre del nodo.
charset	Set de caracteres en el que están codificados los parámetros "name".

9. Nuevas APIS

9.1 Funciones Coshttpdll.dll

Esta dll permite establecer conexiones con servidores web a través del protocolo http.

9.1.1 CosHttpNewRequest

Esta función crea la petición al servidor.

Sintaxis:

```
CosHttpNewRequest() return integer
```

Retorna:

Retorna identificador de la petición.

9.1.2 CosHttpRequestSetUrl

En esta función se indicará la url a la que se van a realizar las peticiones.

Sintaxis:

```
CosHttpRequestSetUrl(requestID as integer, url as char) return integer
```

Parámetros:

requestID	Identificador de la petición.
url	Url del servidor web.

Retorna:

0 si se ha ejecutado correctamente la función.

-1 si no existe el identificador de la petición pasado como parámetro.

9.1.3 CosHttpRequestSetMethod

En esta función se indicará el método que ejecutará la petición.

Métodos de petición implementados: HEAD, GET, POST PUT, DELETE, OPTIONS.

Sintaxis:

```
CosHttpRequestSetMethod(requestID as integer, method as char) return integer
```

Parámetros:

requestID	Identificador de la petición.

Retorna:

0 si se ha ejecutado correctamente la función.

-1 si no existe el identificador de la petición pasado como parámetro.

9.1.4 CoshttpRequestAddHeaderStr

Esta función permite añadir un parámetro a la cabecera de la petición http.

Cada vez que desee añadir un parámetro a la petición http deberá llamarse a esta función.

Sintaxis:

```
CosHttpRequestAddHeaderStr(requestID as integer, headerStr as char) return integer
```

Parámetros:

requestID	Identificador de la petición.
headerStr	Indica el texto del parámetro de la cabecera de la petición http que se va a enviar al servidor.

Retorna:

0 si se ha ejecutado correctamente la función.

-1 si no existe el identificador de la petición pasado como parámetro.

9.1.5 CosHttpRequestSetBody

En esta función se indicará el texto del cuerpo que se enviará al servidor. Este texto se podrá enviar en un objeto char o en un fichero.

Sintaxis:

```
CosHttpRequestSetBody(requestID as integer, bodyStr as char, fromFile as boolean) return integer
```

Parámetros:

requestID	Identificador de la petición.
bodyStr	Cuerpo de la petición. El valor de este parámetro será una cadena de caracteres. Si el valor del parámetro fromFile es FALSE, este parámetro indicará el cuerpo de la petición. Si fromFile es TRUE, indicará la ruta del fichero que contendrá el cuerpo de la petición.
fromFile	Indica si el cuerpo de la solicitud ira en un fichero o en un char.

Retorna:

0 si se ha ejecutado correctamente la función.

-1 si no existe el identificador de la petición pasado como parámetro.

9.1.6 CosHttpSetResponseFile

En esta función se indicará el fichero donde se retornará la respuesta del servidor al recurso solicitado al servidor.

Sintaxis:

```
CosHttpSetResponseFile(requestID as integer, responseFile as char) return integer
```

Parámetros:

requestID	Identificador de la petición.
responseFile	Ruta completa del fichero en el que se retornará la respuesta.

Retorna:

0 si se ha ejecutado correctamente la función.

-1 si no existe el identificador de la petición pasado como parámetro.

9.1.7 CosHttpSetReponseHeaderFile

Función en la que se indicará el fichero donde se retornará el encabezado de la respuesta del servidor.

Sintaxis:

```
CosHttpSetResponseHeaderFile(requestID as integer, responseHeaderFile as char)
return integer
```

Parámetros:

requestID	Identificador de la petición.
responseHeaderFile	Ruta completa del fichero en el que se retornarán los encabezados de la respuesta del servidor.

Retorna:

0 si se ha ejecutado correctamente la función.

-1 si no existe el identificador de la petición pasado como parámetro.

9.1.8 CosHttpSendRequest

Esta función envía al servidor la petición especificada en la función CosHttpRequestSetMethod. Será en este momento cuando el servidor responda y se creen los ficheros indicados en las funciones CosHttpSetResponseFile y CosHttpSetResponseHeaderFile.

Sintaxis:

```
CosHttpSendRequest(requestID as integer, onlyHeaders as boolean) return integer
```

Parámetros:

requestID	Identificador de la petición.
onlyHeaders	Indica si se desea que en la respuesta del servidor se obtenga información de cuerpo y cabeceras o solo cabeceras. Los valores posibles son: TRUE y FALSE. Si el valor es TRUE el servidor no enviará el cuerpo del recurso solicitado, es decir, no se creará el fichero indicado en la función CosHttpSetResponseFile, solo el indicado en CosHttpSetResponseHeaderFile. Este caso solo debe utilizarse cuando se ejecute el método GET. No todos los servidores web implementan la funcionalidad de retornar solamente las cabeceras, por lo que es posible que éste retorne un mensaje de error.

Retorna:

Código de error. En caso de que el proceso se haya ejecutado correctamente retornará 0.

Los códigos de error y sus descripciones son los siguientes:

Código	Descripción	Código	Descripción
0	No ha ocurrido ningún error	46	Código obsoleto
1	Unsupported protocol	47	Number of redirects hit maximum amount
2	Failed initialization	48	An unknown option was passed in to libcurl
3	URL using bad/illegal format or missing URL	49	Malformed telnet option
4	A requested feature, protocol or option was not found built-in in this libcurl due to a build-time decision.	50	Código obsoleto
5	Couldn't resolve proxy name	51	SSL peer certificate or SSH remote key was not OK
6	Couldn't resolve host name	52	Server returned nothing (no headers, no data)
7	Couldn't connect to server	53	SSL crypto engine not found
8	FTP: weird server reply	54	Can not set SSL crypto engine as default
9	Access denied to remote resource.	55	Failed sending data to the peer
10	FTP: The server failed to connect to data port Fallo al conectar al puerto	56	Failure when receiving data from the peer
11	Password desconocida. FTP: unknown PASS reply".	57	Código obsoleto
12	Time out	58	Problem with the local SSL certificate
13	FTP: unknown PASV reply	59	Couldn't use specified SSL cipher
14	Formato 227 respuesta desconocida. FTP: unknown 227 response format	60	Peer certificate cannot be authenticated with given CA certificates
15	FTP: can't figure out the host in the PASV response	61	Unrecognized or bad HTTP Content or Transfer-Encoding
16	Error in the HTTP2 framing layer	62	Invalid LDAP URL
17	FTP: couldn't set file type	63	Maximum file size exceeded
18	Transferred a partial file	64	Requested SSL level failed
19	FTP: couldn't retrieve (RETR failed) the specified file	65	Send failed since rewinding of the data stream failed
20	Código obsoleto	66	Failed to initialise SSL crypto engine
21	Quote command returned error	67	Login denied
22	HTTP response code said error	68	TFTP: File Not Found
23	Failed writing received data to disk/application	69	TFTP: Access Violation
24	Código obsoleto	70	Disk full or allocation exceeded
25	Upload failed (at start/before it took off)	71	TFTP: Illegal operation
26	Failed to open/read local data from file/application	72	TFTP: Unknown transfer ID
27	Out of memory	73	Remote file already exists
28	Timeout was reached	74	TFTP: No such user
29	Código obsoleto	75	Conversion failed
30	FTP: command PORT failed	76	Caller must register CURLOPT_CONV_ callback options
31	FTP: command REST failed	77	Problem with the SSL CA cert (path? access

Código	Descripción	Código	Descripción
			rights?)
32	Código obsoleto	78	Remote file not found
33	Requested range was not delivered by the server	79	Error in the SSH layer
34	Internal problem setting up the POST	80	Failed to shut down the SSL connection
35	SSL connect error	81	Socket not ready for send/recv
36	Couldn't resume download	82	Failed to load CRL file (path? access rights?, format?)
37	Couldn't read a file.	83	Issuer check against peer certificate failed
38	LDAP: cannot bind	84	FTP: The server did not accept the PRET command.
39	LDAP: search failed	85	RTSP CSeq mismatch or invalid CSeq
40	Código obsoleto	86	RTSP session error
41	A required function in the library was not found	87	Unable to parse FTP file list
42	Operation was aborted by an application callback	88	Chunk callback failed
43	A libcurl function was given a bad argumen	89	The max connection limit is reached
44	Código obsoleto	90	SSL public key does not match pinned public key
45	Failed binding local connection end		

9.1.9 CosHttpGetErrorStr

Esta función retorna el texto del mensaje de error de la petición que se ha ejecutado con la función CosHttp-SendRequest.

Sintaxis:

```
CosHttpGetErrorStr (requestID as integer) return Char
```

Parámetros:

requestID Identificador de la petición.

Retorna:

Una cadena de caracteres con el texto del mensaje.

9.1.10 CosHttpGetReturnCode

Esta función retorna el código de estado del protocolo http.

Sintaxis:

```
CosHttpGetReturnCode(requestID as integer) return integer
```

Parámetros:

requestID Identificador de la petición.

Retorna:

El código de estado http.

9.1.11 CosHttpFreeRequest

Libera la petición y toda la memoria que ha usado durante el proceso. Es obligatorio utilizarla cuando termina la conversación.

Sintaxis:

```
CosHttpFreeRequest(requestID as integer) return integer
```

Parámetros:

requestID	Identificador de la petición.

Retorna:

0 si se ha ejecutado correctamente la función.

-1 si no existe el identificador de la petición pasado como parámetro.

9.1.12 CosHttpUseSSL

Esta función indica a la DLL que en la petición pasada como parámetro se utilizará el protocolo SSL.

Sintaxis:

```
CosHttpUseSSL(requestID as integer, useSSL as boolean) return integer
```

Parámetros:

requestID	Identificador de la petición.
useSSL	Booleano que indica si la conexión es segura o no. Sus valores posibles son: TRUE (utiliza protocolo SSL) o FALSE (no utiliza protocolo SSL), siendo éste el valor por defecto.

Retorna:

0 si se ha ejecutado correctamente la función.

-1 si no existe el identificador de la petición pasado como parámetro.

9.1.13 CosHttpIncludeHeaderInResponse

Esta función indica a la DLL que la cabecera de la respuesta del servidor se incluirá en el mismo fichero de respuesta del cuerpo.

Sintaxis:

```
CosHttpIncludeHeaderInResponse(requestID as integer, includeHeader as boolean)
return integer
```

Parámetros:

requestID	Identificador de la petición.
includeHeader	Parámetro que indicará dónde se recibirá la cabecera de la respuesta del servidor. Los valores posibles son: TRUE (cabecera y cuerpo serán retornados en el fichero de respuesta indicado por la función CosHttpSetResponseFile) y

FALSE, que es el valor por defecto y que indica que la cabecera de la respuesta se retornará en el fichero especificado en la función CosHttpSetResponseHeaderFile.

Retorna:

0 si se ha ejecutado correctamente la función.

-1 si no existe el identificador de la petición pasado como parámetro.

9.1.14 CosHttpSetAuthUser

Función en la que se indicará el usuario con el que se valida en el servidor al realizar la petición.

Sintaxis:

```
CosHttpSetAuthUser(requestID as integer, user as char) return integer
```

Parámetros:

requestID	Identificador de la petición.
user	Nombre del usuario.

Retorna:

0 si se ha ejecutado correctamente la función.

-1 si no existe el identificador de la petición pasado como parámetro.

Ver método CosHttpSetAuthMethod.

9.1.15 CosHttpSetAuthPasswd

Función en la que se especifica la contraseña con la que se identifica en el servidor el usuario indicado en la función CosHttpSetAuthUser.

Sintaxis:

```
CosHttpSetAuthPasswd(requestID as integer, passwd as char) return integer
```

Parámetros:

requestID	Identificador de la petición.
passwd	Contraseña con la que se valida el usuario.

Retorna:

0 si se ha ejecutado correctamente la función.

-1 si no existe el identificador de la petición pasado como parámetro.

Ver función CosHttpSetAuthMethod

9.1.16 CosHttpSetAuthMethod

Esta función especifica el tipo de autenticación que se usará en la petición contra el servidor.

Sintaxis:

```
CosHttpSetAuthMethod(requestID as integer, authMethod as integer) return integer
```

Parámetros:

requestID	Identificador de la petición.
authMethod	Tipo de autenticación. Los valores posibles son:
	0 Ninguna.
	1 Basic.
	2 Digest.
	3 GSS-Negotiate.
	4 NTLM.

Retorna:

0 si se ha ejecutado correctamente la función.

-1 si no existe el identificador de la petición pasado como parámetro.

9.1.17 CosHttpSetTimeout

Esta función permite establecer un “timeout” de conexión.

Sintaxis:

```
CosHttpSetTimeout(requestID as integer, secondsTimeout as integer) return integer
```

Parámetros:

requestID	Identificador de la petición.
secondsTimeout	Tiempo en segundos.

Retorna:

0 si se ha ejecutado correctamente la función.

-1 si no existe el identificador de la petición pasado como parámetro.

10. Corrección de errores

10.1 Runtime

- El método LookupColumn de la clase FormTable no mostraba correctamente los valores de los lookup cuando la columna lookup era el resultado de enlazar una tabla con otra y el valor de la variable LOOKUP-DELAYED era TRUE.
- Método ReplaceAt de la clase Char. Si el string de reemplazo es mayor que el string completo no liberaba memoria.
- Error de Cosrun al ejecutar una aplicación Cosmos a través de Terminal Server con sistemas operativos Windows 2012 y Windows 8.
- En las listas de agrupados y agregados en modo compacto, si se cerraban todos los nodos de la lista, al intentar expandir el segundo nodo de ésta había veces que no se abría.

10.2 Cosmos

- No funcionaba el botón OK del método SelectWindow.

10.3 Entorno de desarrollo

- Al pulsar doble clic sobre una palabra en el editor de código, si la palabra tenía el carácter "_" lo tomaba como separador de palabra.
- Code insight. En algunas ocasiones, cuando en el código de la clase Page se utilizaba el code insight con el objeto Self en la clase Page, daba un error de protección general.
- Code insight. Cuando se definía un objeto de la clase Char en la sección variable de la clase Page al utilizar el code insight no mostraba los métodos de esta clase.
- Code insight. Al definir un objeto de la clase Decimal, Smallint o Integer, si se indicaba un método de la clase Numeric sobre el objeto que devolvía el método, el code insight no mostraba la ventana de ayuda con los métodos de la clase.
- Code insight. No se desplegaba la lista de métodos y propiedades para el objeto this.
- Code insight. La lista desplegable se cerraba cuando no existía ningún elemento con el criterio de búsqueda. Esto impedía que se pudiera realizar una nueva búsqueda.
- Code Insight. No mostraba los conversores de la clase.
- En el entorno de desarrollo, cuando se realizaba una búsqueda desde la opción "Find string in Files" de la cadena "a" y las opciones de "Match Whole Word" y "Match Case" estaban seleccionadas, en ocasiones daba un error de protección general.

10.4 CTSQL

- Las funciones LTRIM y RTRIM, no asignaban el tamaño correcto al valor retornado.
- Se ha corregido un error en la versión 3.6 0.36 del motor CTSQL en Windows que provocaba un fallo de protección general cuando se realizaba una query sobre una tabla, poniendo delante el nombre del propietario de la tabla y ese nombre tenía más de 8 caracteres. Este error solamente era reproducible desde una aplicación Java usando el driver JDBC de MultiBase.

10.5 MONITOR

- La aplicación monitor no mostraba con caracteres legibles los nombres de los ficheros bloqueados.