MultiBase Cosmos

Notas a la versión 7.8

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

Índice

1. Implementaciones

1.1 APIS

En esta versión se han implementado dos APIS: COSQRDLL y COSCOMPRESSDLL:

- La DLL COSQRDLL permite generar un archivo de imagen de un código QR.
- La DLL COSCOMPRESSDLL permite crear un archivo comprimido a partir de los ficheros y las carpetas que se le indiquen. También permite descomprimir archivos.

1.1.1 COSSIGNFILE

Se han realizado las modificaciones necesarias para que Cosmos pueda firmar ficheros XML en formato XAdEs.

Además, se ha implementado una nueva función, CosSignFileSignXAdES_Ex, que permite añadir una serie de parámetros extras a la firma.

La versión de API Cossignfile.dll que se distribuye con la versión 7.8 de Cosmos es la versión 1.1.0. Utiliza las librerías de la versión 1.8 de AutoFirma.

1.1.2 TTCHARTDLL

Implementar una función en la API que permita guardar la imagen del chart en un fichero de imagen (BMP, JPG, etc.).

Se ha implementado la función ChartExportImage, que recibe como parámetro el handle de la ventana del chart, el fichero de salida y el tipo de imagen (BMP, JPG).

1.2 Entorno de desarrollo

1.2.1 Editor visual

Se ha implementado la posibilidad de intercambiar la posición entre dos controles, incluido taborder.

1.2.2 Editor de módulos

Se ha implementado la posibilidad de seleccionar/deseleccionar múltiples módulos (includes y librerías) de una lista ordenada alfabéticamente.

1.3 Runtime

1.3.1 Clase JSON

Posibilidad de recorrer un JSON sin necesidad de conocer el nombre de las propiedades. Para ello, se ha implementado el método GetChildList de la clase JSON que retorna un string con el nombre de los hijos de un JSON separados por pipes.

1.3.2 Clase Module

Se ha implementado el método GetClipboardTextUTF8 de la clase Module, que permite obtener texto del portapapeles en utf-8.

Se ha implementado el método LoadJVM que permite cargar una máquina virtual de Java con un CLASSPATH. La implementación de este método permite la utilización de varias APIs que usen clases Java en el mismo proceso *cosrun.exe*.

1.3.3 Conversores de sistemas numéricos

Se han implementado una serie de métodos que permiten convertir un número de un sistema numérico a otro (decimal, hexadecimal y binario).

- En la clase Char se han implementados los métodos: GetSmallinFromCharHex y GetIntegerFromCharHex. Estos métodos permiten obtener el número Smallint o Integer a partir de un string que contiene su representación hexadecimal.
- En la clase Binary se ha implementado el método LoadFromNumber. Este método recibe como parámetro un objeto de las clases Smallint o Integer y lo almacena en el objeto Binary.
    - Si el parámetro que recibe es de tipo Smallint ocupará 2 bytes.
    - Si el parámetro que recibe es de tipo Integer ocupará 4 bytes.
    - Si el parámetro que recibe es de tipo decimal no almacenará nada.
    - Se almacenará en el formato propio del procesador (Little-endian en procesadores Intel x86 compatibles).
- En la clase Numeric se ha implementado el método ToCharHex que permite obtener la representación hexadecimal de un número.

1.3.4 Exportación de imágenes

Se ha implementado un nuevo método de la clase Control, ExportImageEx. Este método, a diferencia del método ExportImage exportará la imagen asociada al control sin que este tenga que estar en primer plano.

2. Nuevos métodos

2.1 Clase Binary

2.1.1 Método LoadFromNumber

Este método permite cargar en un objeto Binary un Smallint o un Integer, similar al método LoadFromChar.

Sintaxis:

```
LoadFromNumber(intval as Numeric)
```

Parámetros:

Intval	Objeto de la clase Smallint o Integer.

Si el objeto es un Smallint ocupará 2 bytes. Si es un Integer ocupará 4 bytes.

Si el objeto que recibe como parámetro es un decimal no almacenará nada.

El formato en el que se almacenará será el del procesador (Little-endian en procesadores Intel x86 compatibles, es decir, el byte de menor peso se almacena en la dirección más baja de memoria y el byte de mayor peso en la más alta).

2.1.2 Método SHA256

Este método permite generar un hash SHA256 de los datos contenidos en el objeto Binary. El hash lo retornará en una cadena alfanumérica de 64 caracteres que representarán 256 bits (32 bytes) en notación hexadecimal.

Sintaxis:

```
SHA256(VAR string as Char)
```

Parámetros:

String	Variable de tipo char pasada por referencia donde retornará el hash SHA256.

2.2 Clase Char

2.2.1 Método GetIntegerFromCharHex

Este método retorna un Integer a partir de su representación hexadecimal almacenada en un objeto Char.

Sintaxis:

```
GetIntegerFromCharHex() return integer
```

2.2.2 Método GetSmallintFromCharHex

Este método retorna un Smallint a partir de su representación hexadecimal almacenada en un objeto Char.

Sintaxis:

```
GetSmallintFromCharHex() return Smallint
```

2.3 Clase Control

2.3.1 Método ExportImageEx

Este método salva el control en un fichero de imagen.

A diferencia del método ExportImage este método exportará la imagen asociada al control sin que el control esté en primer plano.

Sintaxis:

```
ExportImageEx(filePath as Char ,imageType as Char default NULL ,properties as
Char default NULL)
```

Parámetros:

filePath	Ruta donde se guardará la imagen.
imageType	Formato de la imagen. Valores posibles: BMP y JPG.
Properties	Este parámetro puede tener los siguientes valores (separados por un punto y coma ";"): • MONOCHROME: La imagen se genera en blanco y negro. • QUALITY=n: El factor de calidad n puede ser un valor de 0 a 100. Esta propiedad es para el formato JPEG. • ZOOM=n: La imagen se guarda con la ampliación indicada en n.

Este método no se puede utilizar en los controles UserControl.

2.4 Clase Json

2.4.1 Método GetChildList

Este método retorna un string con todos los elementos de un nodo del objeto JSON. Los nombres de los elementos irán separados por el carácter «|». Este método se podrá utilizar para recorrer la estructura de un objeto JSON sin necesidad de conocer previamente el nombre de sus propiedades.

Sintaxis:

```
GetChildList() return char
```

2.5 Clase Module

2.5.1 Método GetClipboardTextUTF8

Este método permite obtener el contenido del portapapeles en formato utf-8.

Sintaxis:

```
GetClipboardTextUTF8() return char
```

2.5.2 Método LoadJVM

Este método permite cargar una máquina virtual de Java con un determinado CLASSPATH.

La máquina virtual cargada será la indicada en las variables COSMOSUSEJAVAVERSION y COSMOSUSELASTJAVAVERSION. Si estas variables no están definidas se cargará la máquina virtual indicada en la variable PATH del sistema.

Sintaxis:

```
LoadJVM(classpath as Char ,loadJarsFolder as Boolean ,loadJavaClasspathFile as Boolean) return integer
```

Parámetros:

Classpath	Ruta completa de los ficheros jar.
loadJarsFoder	Booleano que indica si se deben añadir al classpath los ficheros jar ubicados en el directorio jars del proyecto.
loadJavaClasspathFile	Booleano que indica si se deben añadir al classpath los ficheros jar que se indican en el fichero java.options.

Retorna:

- 0. Si no hay error.
- -1. Si ya existe una máquina virtual de Java cargada para el proceso que se está ejecutando.
- -2. No se ha podido cargar la máquina virtual Java.

2.6 Clase Numeric

2.6.1 Método ToCharHex

Este método permite obtener la representación hexadecimal de un número. El valor se retorna en un objeto Char.

Sintaxis:

```
ToCharHex()return Char
```

Si se ejecuta sobre un Smallint retorna un String de 4 caracteres, 2 por cada byte, con la representación hexadecimal del número.

Si se ejecuta sobre un Integer retorna un String de 8 caracteres, 2 por cada byte, con la representación hexadecimal del número.

Si se ejecuta sobre un decimal retorna un String de al menos 8 caracteres, 2 por cada byte, con la representación hexadecimal del número. Si el número tiene decimales retornará la parte decimal y la parte entera separadas por un punto.

Si el número es negativo retorna la cadena hexadecimal como complemento a 2.

2.7 TTCHARTDLL

2.7.1 Función ChartExportImage

Esta función permite guardar el chart en un fichero de imagen.

Declaración de la función:

```
public dll "ttchartdll.dll" ChartExportImage(hWindow as integer, outFile as
char, fileFormat as char) return boolean
```

Parámetros:

hWindow	Identificador de la ventana.
outFile	Ruta del fichero imagen.
FileFormat	Formato de la imagen. Valores posibles: BMP y JPG.

2.8 API COSSIGNFILE

2.8.1 CosSignFileSignXAdES_Ex

Esta función firmará el fichero XML pasado como parámetro utilizando el formato XAdES.

Declaración de la función:

```
Public dll "CosSignFile.dll" CosSignFileSignXAdES_Ex (
    signID as integer,
    inFilePath as char,
    outFilePath as char,
    keystorePath as char,
    keystoreFormat as char,
    keyStorePasswd as char,
    alias as char,
    aliasPasswd as char,
    algorithm as char,
    format as char,
    mode as char,
extraParams as char)
 return Integer
```

Parámetros:

signID	Identificador de la firma retornado en la llamada a la función CosSignFileCreateSigner.
inPDFPath	Ruta absoluta del fichero XML que se desea firmar.
outPDFPath	Ruta absoluta del fichero XML de salida resultado de la firma del fichero XML indicado en el parámetro “inPDFPath”.
keystorePath	Ruta absoluta del fichero de almacén de certificados donde se encuentra almacenado el certificado que se desea utilizar para realizar el proceso de firma.
keystoreFormat	Cadena de caracteres que indica el formato del almacén de certificados. Los posibles valores de este parámetro se podrán obtener invocando la función CosSignFileGetKeystoreFormats (jks, pkcs12, etc.).
keystorePasswd	Contraseña del almacén de certificados especificado en el parámetro “keystorePath”.

alias	Alias del certificado electrónico que se desea utilizar para realizar la firma del archivo.
aliasPasswd	Contraseña del certificado electrónico especificado en el parámetro “alias”.
algorithm	Algoritmo de cifrado que se desea utilizar. La lista de algoritmos soportados se podrá consultar ejecutando la función CosSignFileGetSignAlgorithms (SHA1withRSA, SHA256withRSA, etc.).
format	Formato de firma. La lista de formatos de firma soportados se podrá consultar ejecutando la función CosSignFileGetPAdESSignFormats.
mode	Modo de firma. La lista de modos de firma soportados se podrá consultar ejecutando la función CosSignFileGetSignModes (implícito, explícito, etc.).
extraParams	Lista de parámetros adicionales. Los posibles valores son: policyIdentifier, policyIdentifierHash, policyIdentifierHashAlgorithm, policyDescription y policyQualifier. El formato es: clave:valor

```
extraParams = '"policyIdentifier":"X.XX.XXX.X.X.X.X.X.X.X"'
    + ";"
    + '"policyIdentifierHash":"V8lVVNGDCPen6VELRD1Ja8HARFk="'
    + ";"
    + '"policyIdentifierHashAlgorithm":"SHA-256"'
    + ";"
    + '"policyDescription":"Firma electronica"'
    + ";"
    + '"policyQualifier":"http://paginaweb"';
```

3. Correcciones

3.1 Runtime

- Método GetClipBoardText. Al ejecutar este método, el portapapeles del sistema operativo quedaba inhabilitado mientras no terminará el proceso *cosrun.exe*.
- Impresión a PDF. El control variable no se imprimía en vertical.
- Método Allowcolumnheaderfilter. Listas de tipo sql.
    - Con más de una tabla no se aplicaba el filtro si el nombre de la columna que este método recibía como parámetro estaba presente en más de una tabla.
    - Si la sentencia contenía la cláusula limit, al filtrar por la cabecera la instrucción que Cosmos generaba no era correcta y no se retornaba ningún registro.
    - Si la lista tenía muchos registros, al filtrar por una columna daba error de protección general y la aplicación dejaba de funcionar.
- Exportación a PDF. No exportaba correctamente un listado que utilizaba el Font Bancaixa128Ch.
- Método LoadFromFile de la clase Char. Si no existía el fichero a cargar o no era posible cargarlo, asignaba "" (comillas comillas) a la variable en lugar de asignarle NULL.
- Método GetWindowsVersion. En el caso de Windows 10 no retornaba correctamente el valor.
- Ejecución de funciones/métodos Java. No se limpiaba correctamente el buffer de excepciones.
- El Método ExportToExcel retornaba siempre el valor null en lugar de un booleano.

3.2 Repositorio

- Al convertir un repositorio en XML e importarlo a CRF las columnas de tipo enumerado o boolean no se creaban correctamente. En el archivo con formato CRF, aunque se almacenaban correctamente los valores, no almacenaba el número de valores. Esto hacía que el compilador no funcionara correctamente.
- Al exportar un repositorio de CRF a XML no exportaba los “VALUES” y “TAGS” cuando el tipo estaba marcado como “Generic”.

3.3 Editor de código

- Code Insight.
    - Clase JSON. Al escribir el carácter “.” después de un método de la clase JSON no mostraba los métodos y/o propiedades que exporta la función de retorno.
    - Clase MENU. Al escribir el carácter "." después del método *Option* de la clase Menú, en la sección code de la clase se producía un error de protección general y se detenía la ejecución de la aplicación “Cosmos Visual Editor”.

3.4 APIS.

- COSSINGFILE. No firmaba correctamente en formato XAdEs.