Kaseya 2

Network Monitor API

Guía del usuario

Versión R8

Español

November 5, 2014

Agreement

The purchase and use of all Software and Services is subject to the Agreement as defined in Kaseya's "Click-Accept" EULATOS as updated from time to time by Kaseya at http://www.kaseya.com/legal.aspx. If Customer does not agree with the Agreement, please do not install, use or purchase any Software and Services from Kaseya as continued use of the Software or Services indicates Customer's acceptance of the Agreement."

Contenido

API de Lua para Network Monitor

En esta documentación, se abarca la API de Lua para **Network Monitor**. **Network Monitor** usa Lua 5.0.

Lua

Lua es un lenguaje de programación eficaz y liviano diseñado para extender aplicaciones. Además, se utiliza con frecuencia como lenguaje independiente de uso general. Lua es software gratuito que combina una sintaxis de procedimiento simple con construcciones eficaces de descripción de datos basadas en matrices asociativas y en una semántica extensible. Lua se escribe en forma dinámica, se interpreta a partir de códigos de byte y tiene administración de memoria automática con recolección de elementos no utilizados, lo que lo hace ideal para configurar, generar scripts y crear prototipos con rapidez.

Nota: En esta documentación, no se abarca el lenguaje Lua. Para obtener más información sobre el lenguaje Lua, visite el sitio **http://www.lua.org.** *(http://www.lua.org.)*

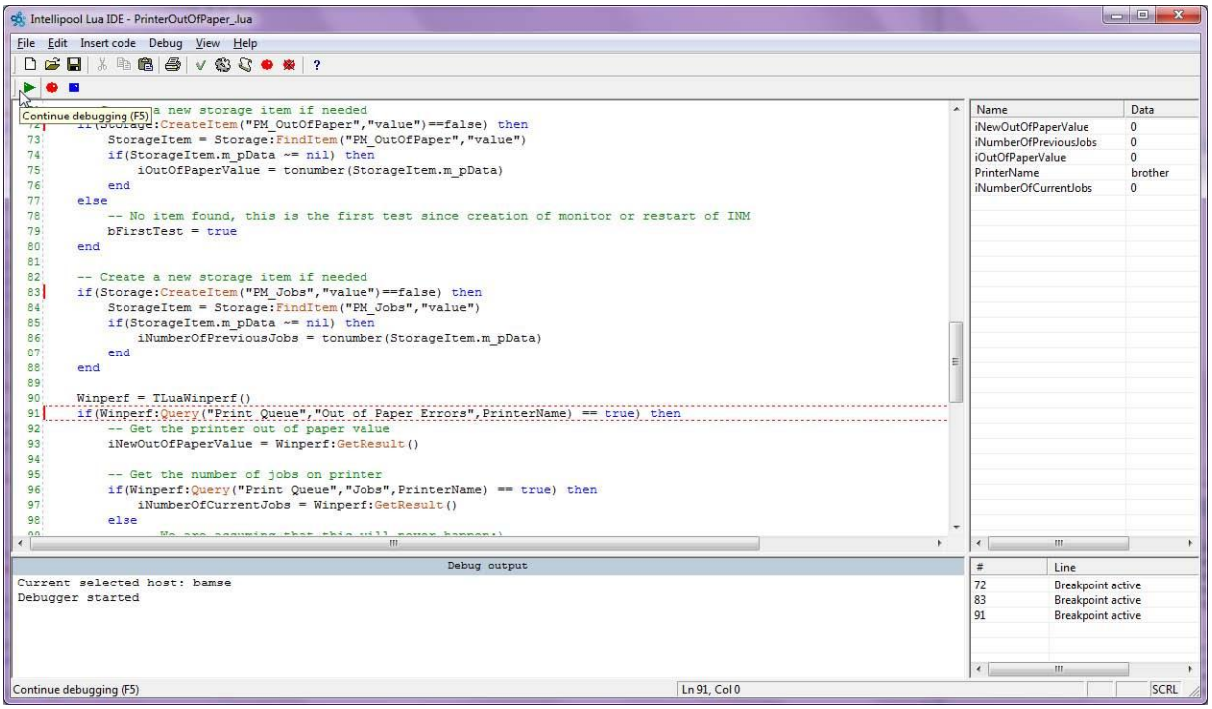
Network Monitor y Lua

Los clientes pueden usar el lenguaje de script Lua para crear monitores personalizados para probar los sistemas y equipos no admitidos por ninguna solución de supervisión actual.

En el entorno de desarrollo que proporciona Kaseya, se pueden crear y probar nuevos monitores, acciones y eventos antes de exportarlos y usarlos en Kaseya Network Monitor.

Los desarrolladores pueden consultar una amplia biblioteca de clases prefabricadas, como clientes SFTP, clientes HTTP y administración de archivos.

El entorno de desarrollo incluye las características de depurador, resaltado de palabras clave, ayuda integrada, entre otras, disponibles en las herramientas de desarrollo más avanzadas. El entorno de desarrollo se puede descargar de nuestra página principal en **http://www.kaseya.com** *(http://www.kaseya.com)*.



IDE Lua v3

Capítulo 1

Modelo de programación

Cuando se crea un script Lua personalizado para usar con Kaseya Network Monitor, el script debe cumplir una cantidad de requisitos para que Kaseya Network Monitor lo pueda ejecutar correctamente.

En este capítulo

Script avanzado

El modelo de script avanzado le brinda al autor nuevas herramientas eficaces para controlar los parámetros que se dan como argumentos al script. Esto permite que se creen scripts Lua que tengan el mismo aspecto que los tipos de monitores nativos.

Nombres de función reservados

Existen dos nombres de función reservados, que utiliza **Network Monitor** para consultar información. Estos nombres de función no se pueden utilizar para ningún otro propósito.

OnConfigure

Network Monitor invoca esta función para que el script rellene una instancia de clase LuaScriptConfigurator. La información se utiliza luego para crear una interfaz de usuario para el script. El nombre de la instancia debe ser “Config” (tenga en cuenta la mayúscula) para que **Network Monitor** pueda encontrarla en la pila de Lua cuando se devuelva la función.

OnEnumerate

Se pueden enumerar todos los campos en la interfaz de usuario. **Network Monitor** invoca la función OnEnumerate para que el script rellene una estructura de datos, LuaScriptEnumResult, con valores que selecciona el usuario. La función OnEnumerate tiene un parámetro, sFieldToEnum, que el script usa para determinar a qué campo o argumento proporcionarle los resultados de la enumeración. La instancia devuelta se debe denominar “Enum” (tenga en cuenta la mayúscula).

El punto de entrada

El modelo de script avanzado requiere que la función OnConfigure establezca el nombre de la función de punto de entrada. **Network Monitor** invoca esta función para iniciar la ejecución del script. De manera predeterminada, el nombre del punto de entrada es “main”, pero el programador puede establecer cualquier nombre, excepto los nombres de función reservados.

Ejemplo

```
--This function is called by KNM when enumerating a field

function OnEnumerate(sFieldToEnum)

  --The variable returned must be called "Config" so KNM can find it.
  Enum = LuaScriptEnumResult()

  --Second argument
  if sFieldToEnum == "Argument 2" then
        Enum:Add("First value")
        Enum:Add("Second value")
        Enum:Add("Third value") end
  return Enum
end

--This function is called by KNM to retrieve a script configuration

function OnConfigure()

  --The variable returned must be called "Config" so KNM can find it.
  Config = LuaScriptConfigurator()

  --Author.
  Config:SetAuthor("My name")

  --Description.
  Config:SetDescription("Description of the script, including usage,
parameters etc")

  --Minimum build version of KNM, set to zero for if no specificbuild version
is required.
  Config:SetMinBuildVersion(0)

  --Script version (major/minor)
  Config:SetScriptVersion(1,0)

  --A parameter configuration, add them in the order the script isextracting
them.
  Config:AddArgument("Argument 1","This is the description of thefirst
argument",LuaScriptConfigurator.CHECK_NOT_EMPTY)

  --Add another parameter, a select box with 3 values.
  Config:AddArgument("Argument 2","This is the description of thesecond
argument",LuaScriptConfigurator.CHECK_NOT_EMPTY+LuaScriptConfigurator.ENUM
_AVAIL)

  --Set the entry point, this is the function called by KNM
Config:SetEntryPoint("main")

  --Done with configuration, return the asset
  return Config
end

--This is the entry point
```

```
function main()

  sFirstArgument = GetArgument(0)
  sSecondArgument = GetArgument(1)

  SetExitStatus("OK",true)

end
```

Script simple

El modelo de script simple se utilizó en **Network Monitor** desde la primera versión, y ahora debe considerarse obsoleto. Se mantiene por cuestiones de compatibilidad con scripts más antiguos.

Contexto de activos

Las funciones son relativas al contexto de los activos.

Todas las llamadas que acceden a recursos son relativas al activo primario. Por ejemplo, si el script abre un archivo, la ruta que se proporciona a la función de apertura debe ser relativa al activo.

Ejemplo

Establezca la dirección del equipo Windows “servidordedominio” como host.

```
TLuaFile:Open("C:\\test.txt");
```

Al invocar la función, el script abre el archivo test.txt ubicado en el disco duro C: de la computadora “servidordedominio”.

Esta también es la razón por la cual todas las clases relacionadas con la comunicación, como TLuaFTPClient, TLuaHTTPClient y TLUASocket, sólo tienen un argumento de número de puerto; el marco codifica la dirección IP de forma rígida en el activo actual.

Resultado

Cuando se produce la salida de un script, este necesita comunicarle a **Network Monitor** si la prueba fue correcta o no. Se suministra una función global para este propósito, SetExitStatus. SetExitStatus es una función obligatoria, que se debe invocar antes de que termine el script.

Capítulo 2

Funciones globales

Las funciones globales son funciones que no están relacionadas con un activo. Existe una cantidad de funciones globales en la API de Lua para **Network Monitor**. Algunas deben invocar cuando se produce la salida de un script.

En este capítulo

ConvertFromUTF16

string ConvertFromUTF16(local UTF16data,int iSize)

Valores de devolución

Una cadena de 8 bits convertida a partir de la cadena UTF16.

Parámetros

- UTF16data: cadena UTF16 (doble byte) leída por TLuaFile::ReadData.
- iSize: tamaño de la cadena.

Comentarios

La función sólo acepta datos creados por la función TLuaFile::ReadData.

FormatErrorString

string FormatErrorString(int iError)

Valores de devolución

Una cadena que describe el código de error iError.

Parámetros

- iError: un código de error de Windows que se obtuvo anteriormente al invocar la función GetLastError.

Comentarios

Esta función puede utilizarse para proporcionarle al usuario texto significativo en lugar de un código de error.

GetArgument

string GetArgument(int iNumber)

Valores de devolución

Un argumento transmitido por la aplicación de llamada.

Parámetros

- iNumber: un índice de base cero del argumento que se debe recuperar. La cantidad máxima de argumentos se puede determinar invocando GetArgumentCount.

Comentarios

Una aplicación de llamada puede transmitir una cantidad de argumentos al script Lua para personalizar su comportamiento. Con esta función y la GetArgumentCount relacionada, el programador puede extraer los argumentos.

GetArgumentCount

int GetArgumentCount()

Valores de devolución

La cantidad de argumentos que la aplicación de llamada transmite al programa.

Comentarios

Una aplicación de llamada puede transmitir una cantidad de argumentos al script Lua para personalizar su comportamiento. Con esta función, el programador puede determinar la cantidad de argumentos que hay para extraer.

GetLastError

int GetLastError()

Valores de devolución

El último código de error generado por invocación a una función de biblioteca. El código de error es un código de error estándar de Windows.

Comentarios

La función SetLastError se puede usar para borrar el código de error de Windows actual antes de invocar una función .sds.

GetDeviceAddress

string GetDeviceAddress()

Valores de devolución

La dirección introducida en el campo de dirección del dispositivo.

Comentarios

La cadena se puede usar como identificador único en el momento de guardar datos en TLuaStorage.

IsIDE

bool IsIDE()

Valores de devolución

El valor booleano es verdadero si el IDE ejecuta el script y es falso si **Network Monitor** ejecuta el script.

Comentarios

Se pueden utilizar estas funciones si **Network Monitor** o el IDE ejecutan el script.

MessageBox

MessageBox(string sText)

Parámetros

- sText: texto que se muestra en el cuadro de mensaje.

Comentarios

Esta función invoca que se muestre una cadena en un cuadro de mensaje estándar del SO. Esta función sólo está disponible en el IDE. Observe que, cuando se muestra el cuadro de mensaje, se frena la ejecución del script hasta que se cierra.

print

print(string sText)

Parámetros

- sText: texto que se imprime en la ventana de salida.

Comentarios

Esta función se puede utilizar para imprimir texto en la ventana de salida con fines de depuración. Cuando **Network Monitor** ejecuta el script, el texto impreso con esta función no cumple ningún propósito.

SetExitStatus

SetExitStatus(string sString,bool bSuccess)

Parámetros

- sString: una cadena que describe el resultado del script.
- bSuccess: si es distinto de cero (el valor booleano es verdadero), se considera que el marco ejecutó el script correctamente. Si este valor está establecido en cero (el valor booleano es falso), también se debe invocar la función SetErrorString con una cadena que describa el estado del error.

Comentarios

Se debe invocar esta función cuando se produce la salida de un script. La función comunica a **Network Monitor** si el script se realizó correctamente o no. Si el script se ejecuta en el contexto de un agente, **Network Monitor** utiliza el texto proporcionado con la función para establecer el texto del último estado en la interfaz.

SetLastError

SetLastError(int iErrorCode)

Parámetros

- iErrorCode: un valor entero que corresponde a un código de error específico de Windows.

Comentarios

Esta función establece el último código de error que GetLastError puede recuperar más adelante.

StoreStatisticalData

bool StoreStatisticalData(int iRecordSet,float fData,float fThreshold,string Unit)

Valores de devolución

Resultado verdadero si los datos se almacenaron correctamente en la base de datos estadística; resultado falso si hay un error de parámetro.

Parámetros

- iRecordSetIndex: un índice de base cero del canal estadístico en el que se almacenan los datos. Consulte los comentarios para obtener constantes válidas.
- fData: datos de punto flotante que muestrea el script.
- fThreshold: valor de umbral optativo para los datos de ejemplo. Este valor debe ser constante en todas las llamadas.
- Unit: cadena optativa que describe la unidad de los datos. Este valor debe ser constante en todas las llamadas. La cadena puede tener 16 caracteres como máximo. De lo contrario, la invocación falla.

Comentarios

Esta función le da al script la capacidad de almacenar datos estadísticos. Actualmente, existen 8 canales que se pueden utilizar con este fin. El parámetro iRecordSetIndex puede ser una de las siguientes constantes.

```
LUA_RECORDSET_1
LUA_RECORDSET_2
LUA_RECORDSET_3
LUA_RECORDSET_4
LUA_RECORDSET_5
LUA_RECORDSET_6
LUA_RECORDSET_7
LUA_RECORDSET_8
```

StoreStatisticalData

bool StoreStatisticalData(int iRecordSet,float fData,float fThreshold,int iVirtualType,int iVirtualUnit,string Unit)

Valores de devolución

Resultado verdadero si los datos se almacenaron correctamente en la base de datos estadística; resultado falso si hay un error de parámetro.

Parámetros

- iRecordSetIndex: un índice de base cero del canal estadístico en el que se almacenan los datos. Consulte los comentarios para obtener constantes válidas.
- fData: datos de punto flotante que muestrea el script.
- fThreshold: valor de umbral optativo para los datos de ejemplo. Este valor debe ser constante en todas las llamadas.
- iVirtualType: tipo de datos almacenados.

- iVirtualUnit: unidad seleccionada del tipo almacenado. Consulte los comentarios para obtener combinaciones válidas de tipos y unidades.
- Unit: cadena optativa que describe la unidad de los datos. Este valor debe ser constante en todas las llamadas. La cadena puede tener 16 caracteres como máximo. De lo contrario, la invocación falla.

Comentarios

Esta función sólo está disponible para scripts avanzados. La diferencia entre esta función y la función anterior con el mismo nombre es la capacidad de almacenar información de tipo con los datos.

iVirtualType e iVirtualUnit se pueden utilizar en las siguientes combinaciones:

```
VT_SWAP_UTILIZATION
VT_MEMORY_UTILIZATION
VT_DISK_UTILIZATION
VT_CPU_UTILIZATION
  UNIT_TYPE_PERCENT

VT_FREE_DISKSPACE
  UNIT_TYPE_MEGABYTE
  UNIT_TYPE_GIGABYTE
  UNIT_TYPE_TERABYTE

VT_SQL_QUERY
  UNIT_TYPE_NONE

VT_TEMPERATURE:
  UNIT_TYPE_FAHRENHEIT
  UNIT_TYPE_CELSIUS
  UNIT_TYPE_KELVIN

VT_HUMIDITY
  UNIT_TYPE_PERCENT

VT_WETNESS
  UNIT_TYPE_NONE

VT_VOLTAGE
  UNIT_TYPE_VOLT

VT_BANDWIDTH_UTILIZATION
  UNIT_TYPE_PERCENT

VT_BANDWIDTH_USAGE
  UNIT_TYPE_KBPS
  UNIT_TYPE_MBPS
  UNIT_TYPE_GBPS

VT_DIRECTORY_SIZE:
  UNIT_TYPE_MEGABYTE
  UNIT_TYPE_GIGABYTE
  UNIT_TYPE_TERABYTE

VT_DIRECTORY_COUNT
  UNIT_TYPE_NONE

VT_PING_ROUNDTRIP
  UNIT_TYPE_MILLISECONDS
  UNIT_TYPE_SECONDS

VT_PING_PACKETLOSS
  UNIT_TYPE_PERCENT

VT_MAIL_ROUNDTRIP:
  UNIT_TYPE_MILLISECONDS
  UNIT_TYPE_SECONDS
```

```
VT_MEMORY_USAGE
  UNIT_TYPE_MEGABYTE
  UNIT_TYPE_GIGABYTE

VT_TRANSFER_SPEED
  UNIT_TYPE_NONE

VT_HTTP_FETCHTIME
  UNIT_TYPE_MILLISECONDS
  UNIT_TYPE_SECONDS

VT_WMI_GENERIC_VALUE

VT_LUA_GENERIC_VALUE

VT_WINPERF_GENERIC_VALUE

VT_SSH2SCRIPT_GENERIC_VALUE

VT_SNMP_GENERIC_VALUE
  UNIT_TYPE_NONE

VT_CURRENT
  UNIT_TYPE_AMPERE

VT_FANSPEED
  UNIT_TYPE_RPM

VT_LUMINOSITY
  UNIT_TYPE_LUX
```

El parámetro iRecordSetIndex puede ser una de las siguientes constantes.

```
LUA_RECORDSET_1
LUA_RECORDSET_2
LUA_RECORDSET_3
LUA_RECORDSET_4
LUA_RECORDSET_5
LUA_RECORDSET_6
LUA_RECORDSET_7
LUA_RECORDSET_8
```

Wait

Wait(int iMs)

Parámetros

- iMs: la cantidad de milisegundos que debe esperar la ejecución del script.

Comentarios

Esta función invoca que la función de “suspensión” del SO suspenda la ejecución del subproceso que ejecuta al script.

Capítulo 3

LuaScriptEnumResult

Esta clase proporciona una interfaz para introducir resultados de enumeración en la función de devolución de llamada de OnEnumeration.

En este capítulo

Script de ejemplo: OnEnumerate

```
function OnEnumerate(sFieldToEnum)

  --The variable returned must be called "Config" so KNM can find it.
  Enum = LuaScriptEnumResult()

  --Second argument
  if sFieldToEnum == "Argument 2" then
        Enum:Add("First value")
        Enum:Add("Second value")
        Enum:Add("Third value")
  end

  return Enum
end
```

Agregar

Add(const string &sDisplayValue,const string &sUsageValue="")

Parámetros

- sDisplayValue: valor para mostrar como opción para seleccionar.
- sUsageValue: (optativo) un valor que se utiliza en lugar del valor para mostrar.

Comentarios

El valor sUsageValue se puede usar cuando se tienen valores muy complejos y extensos, y es necesario mostrar las opciones de manera más simple. Cuando se lo utiliza, el valor sDisplayValue es el valor que se presenta al usuario, pero el valor sUsageValue es el valor que usa **Network Monitor**.

Capítulo 4

LuaScriptConfigurator

Esta clase proporciona una interfaz para crear información de configuración que **Network Monitor** usa para presentar una interfaz de usuario para el script.

En este capítulo

Script de ejemplo: OnConfigure

```
function OnConfigure()

  --The variable returned must be called "Config" so KNM can find it.
  Config = LuaScriptConfigurator()

  --Author.
  Config:SetAuthor("My name")

  --Description.
  Config:SetDescription("Description of the script, including usage,
parameters etc")

  --Minimum build version of KNM, set to zero for if no specificbuild version
is required.
  Config:SetMinBuildVersion(0)

  --Script version (major/minor)
  Config:SetScriptVersion(1,0)

  --A parameter configuration, add them in the order the script is extracting
them.
  Config:AddArgument("Argument 1","This is the description of thefirst
argument",LuaScriptConfigurator.CHECK_NOT_EMPTY)

  --Add another parameter, a select box with 3 values.
  Config:AddArgument("Argument 2","This is the description of thesecond
argument",LuaScriptConfigurator.CHECK_NOT_EMPTY+LuaScriptConfigurator.ENUM
_AVIL)

  --Set the entry point, this is the function called by KNM
  Config:SetEntryPoint("main")

  --Done with configuration, return the asset
  return Config
end
```

AddArgument

int AddArgument(string sName,string sDescription,int iFlags);

Valores de devolución

Un identificador que se puede utilizar para referirse a este argumento en llamadas subsiguientes.

Parámetros

- sName: nombre del campo de argumento.
- sDesc: descripción del campo iFlags (indicadores de control de validación). Consulte los comentarios para obtener información sobre los indicadores.

Comentarios

Estos son los indicadores válidos. Algunos se pueden combinar.

CHECK_NOTHING	Se aceptan valores predeterminados de cualquier tipo, incluso la ausencia de texto.
CHECK_NOT_EMPTY	Comprueba si el argumento está vacío. No se puede combinar con CHECK_NOTHING.
CHECK_RANGE_LOW	Se debe utilizar con CHECK_NUMERIC. Valida que el valor numérico esté dentro del intervalo (intervalo bajo).
CHECK_RANGE_HIGH	Se debe utilizar con CHECK_NUMERIC. Valida que el valor numérico esté dentro del intervalo (intervalo alto).
CHECK_NUMERIC	Valida que el valor sea numérico (real o entero).
ENUM_AVAIL	Indica que hay una devolución de llamada de enumeración con valores predefinidos disponible para este campo.

SetCharacterLimits

SetCharacterLimits(int iArgIndex,int iMaxCharacters,int iMaxVisibleCharacters)

Parámetros

- iArgIndex: el identificador devuelto por AddArgument.
- iMaxCharacters: cantidad máxima de caracteres de entrada para el argumento.
- iMaxVisibleCharacters: la cantidad máxima de caracteres visibles. Debe ser igual o inferior a iMaxCharacters.

Comentarios

La función establece la longitud máxima de un argumento y cuántos de esos caracteres son visibles en la interfaz (longitud del campo de entrada).

SetNumericLimits

SetNumericLimits(int iArgIndex,float fLow,float fHigh)

Parámetros

- iArgIndex: el identificador devuelto por **AddArgument** *(página 18)*.
- Low: intervalo bajo
- High: intervalo alto

Comentarios

Esta función establece el intervalo aceptable de los valores reales y enteros introducidos en el campo. El argumento debe tener establecidos los indicadores CHECK_RANGE_LOW y CHECK_RANGE_HIGH.

SetEntryPoint

SetEntryPoint(string sName)

Parámetros

- sName: nombre de la función de punto de entrada.

Comentarios

La función registra el nombre de la función de punto de entrada. Esta es la función que **Network Monitor** invoca como punto de partida de la ejecución. El valor predeterminado es “main”.

SetAuthor

SetAuthor(string sName)

Parámetros

- sName: nombre del autor del script.

Comentarios

Esta función establece el autor del script. Se utiliza para fines descriptivos cuando un usuario carga un script de terceros, a fin de informarle quién escribió el script.

SetDescription

SetDescription(string sDescription)

Parámetros

- sDescription: una descripción de la función del script.

Comentarios

Mediante la descripción del script, se le debe informar al usuario —en pocas líneas— cuál es la función del script y si este tiene limitaciones conocidas. No hay un límite máximo de texto, pero debe ser breve.

SetMinBuildVersion

SetMinBuildVersion(int iMinBuildNumber)

Parámetros

- iMinBuildNumber: el número de versión de **Network Monitor** que el script requiere como mínimo.

Comentarios

El número de versión mínima es un campo muy importante para establecer. Este comunica a **Network Monitor** si el script se puede utilizar con la versión actual de **Network Monitor**. De manera predeterminada, este número se debe establecer en el número de la versión que utilizó el autor para probar el script.

SetScriptVersion

SetScriptVersion(int iMajor,int iMinor)

Parámetros

- iMajor: el número de versión principal del script.

- iMinor: el número de versión secundaria del script.

Comentarios

El autor del script debe establecer un número de versión del script. Una versión principal de 0 indica que el script está en una etapa “beta” y que otros usuarios sólo lo deben utilizar para continuar desarrollándolo. Cada vez que se modifica el script, se debe aumentar el número de la versión. Un cambio en el número de versión principal debe reflejar una rescritura o una mejora de gran escala. El número de versión secundaria indica una mejora de menor escala.

Capítulo 5

TLuaDateTime

TLuaDateTime proporciona funciones de fecha y hora. “Time” es la hora local representada en segundos desde el 1 de enero de 1970.

En este capítulo

Agregar

Add(TLuaDateTime DateTime)

Parámetros

- DateTime: instancia de TLuaDateTime que se obtiene de otra función de clase o se construye.

Comentarios

Esta función agrega al activo la hora contenida en el parámetro DateTime.

Crear

Create(int iYear,int iMonth,int iDay,int iHour,int iMinute,int iSecond)

Parámetros

- iYear: año; por ejemplo, 1972.
- iMonth: número de mes; por ejemplo, 10.
- iDay: número de día del mes; por ejemplo, 2.
- iHour: hora para utilizar; puede ser cero.
- iMinute: minutos para utilizar; puede ser cero.
- iSecond: segundos para utilizar; puede ser cero.

Comentarios

Esta función crea un valor TLuaDateTime que contiene un tiempo absoluto.

CreateSpan

CreateSpan(int iHour,int iMinute,int iSecond)

Parámetros

- iHour: hora para utilizar; puede ser cero.
- iMinute: minutos para utilizar; puede ser cero.
- iSecond: segundos para utilizar; puede ser cero.

Comentarios

Esta función crea un valor TLuaDateTime que no contiene un tiempo absoluto, sino un lapso de tiempo que se puede utilizar para sumar a otro activo de TLuaDateTime o restar de este.

Igual a

bool Equal(TLuaDateTime DateTime)

Valores de devolución

El resultado es verdadero si DateTime es igual; de lo contrario, es falso.

Parámetros

- DateTime: instancia de TLuaDateTime que se obtiene de otra función de clase o se construye.

Get

int Get()

Valores de devolución

Cantidad de segundos que contiene la instancia.

Comentarios

Esta función se puede usar para recuperar la cantidad de segundos que contiene la instancia, en forma de tiempo absoluto.

GetDate

string GetDate(string sFormat=NULL)

Valores de devolución

Devuelve una cadena con la hora actual, con formato según el parámetro sFormat o en el formato predeterminado.

Parámetros

- sFormat: cadena optativa que contiene un formato alternativo de la hora devuelta.El formato predeterminado es AA-MM-DD. Consulte la sección de comentarios para obtener información sobre los indicadores que se pueden usar.

Comentarios

Devuelve una cadena con la hora contenida por la instancia. El formato predeterminado es AA-MM-DD. Si proporciona su propio código de formato, puede modificar la forma en que se devuelve la hora.

Indicadores de formato

- %a: nombre de día de semana abreviado.
- %A: nombre de día de semana completo.
- %b: nombre de mes abreviado.
- %B: nombre de mes completo.
- %c: representación de fecha y hora adecuada para la configuración regional.
- %d: el día del mes como número decimal (de 01 a 31).
- %H: la hora en formato de 24 horas (de 00 a 23).
- %I: la hora en formato de 12 horas (de 01 a 12).
- %j: el día del año como número decimal (de 001 a 366).
- %m: el mes como número decimal (de 01 a 12).
- %M: los minutos como número decimal (de 00 a 59).
- %p: indicador actual de a. m./p. m. de la configuración regional para el reloj de 12 horas.
- %S: los segundos como número decimal (de 00 a 59).
- %U: la semana del año como número decimal, tomando el domingo como primer día de la semana (de 00 a 53).
- %w: el día de la semana como número decimal (de 0 a 6; el domingo es 0).

- %W: la semana del año como número decimal, tomando el domingo como primer día de la semana (de 00 a 53).
- %x: representación de fecha para la configuración regional actual.
- %X: representación de hora para la configuración regional actual.
- %y: el año sin el siglo, como número decimal (de 00 a 99).
- %Y: el año con el siglo, como número decimal.
- %z, %Z: nombre o abreviatura de la zona horaria; sin caracteres si no se conoce la zona horaria.

GetTime

string GetTime(string sFormat=NULL)

Valores de devolución

Devuelve una cadena con la hora actual, con formato según el parámetro sFormat o en el formato predeterminado.

Parámetros

- sFormat: cadena optativa que contiene un formato alternativo de la hora devuelta. El formato predeterminado es HH:MM:SS. Consulte la sección de comentarios para obtener información sobre los indicadores que se pueden usar.

Comentarios

Devuelve una cadena con la hora contenida por la instancia. El formato predeterminado es HH-MM-DD. Si proporciona su propio código de formato, puede modificar la forma en que se devuelve la hora.

Indicadores de formato

- %a: nombre de día de semana abreviado.
- %A: nombre de día de semana completo.
- %b: nombre de mes abreviado.
- %B: nombre de mes completo.
- %c: representación de fecha y hora adecuada para la configuración regional.
- %d: el día del mes como número decimal (de 01 a 31).
- %H: la hora en formato de 24 horas (de 00 a 23).
- %I: la hora en formato de 12 horas (de 01 a 12).
- %j: el día del año como número decimal (de 001 a 366).
- %m: el mes como número decimal (de 01 a 12).
- %M: los minutos como número decimal (de 00 a 59).
- %p: indicador actual de a. m./p. m. de la configuración regional para el reloj de 12 horas.
- %S: los segundos como número decimal (de 00 a 59).
- %U: la semana del año como número decimal, tomando el domingo como primer día de la semana (de 00 a 53).
- %w: el día de la semana como número decimal (de 0 a 6; el domingo es 0).
- %W: la semana del año como número decimal, tomando el domingo como primer día de la semana (de 00 a 53).
- %x: representación de fecha para la configuración regional actual.
- %X: representación de hora para la configuración regional actual.
- %y: el año sin el siglo, como número decimal (de 00 a 99).
- %Y: el año con el siglo, como número decimal.

- %z, %Z: nombre o abreviatura de la zona horaria; sin caracteres si no se conoce la zona horaria.

Greater

bool Greater(TLuaDateTime DateTime)

Valores de devolución

El resultado es verdadero si DateTime es menor; de lo contrario, es falso.

Parámetros

- DateTime: instancia de TLuaDateTime que se obtiene de otra función de clase o se construye.

GreaterOrEqual

bool GreaterOrEqual(TLuaDateTime DateTime)

Valores de devolución

El resultado es verdadero si DateTime es menor o igual; de lo contrario, es falso.

Parámetros

- DateTime: instancia de TLuaDateTime que se obtiene de otra función de clase o se construye.

Less

bool Less(TLuaDateTime DateTime)

Valores de devolución

El resultado es verdadero si DateTime es mayor; de lo contrario, es falso.

Parámetros

- DateTime: instancia de TLuaDateTime que se obtiene de otra función de clase o se construye.

LessOrEqual

bool LessOrEqual(TLuaDateTime DateTime)

Valores de devolución

El resultado es verdadero si DateTime es mayor o igual; de lo contrario, es falso.

Parámetros

- DateTime: instancia de TLuaDateTime que se obtiene de otra función de clase o se construye.

No igual a

bool NotEqual(TLuaDateTime DateTime)

Valores de devolución

El resultado es verdadero si DateTime no es igual; de lo contrario, es falso.

Parámetros

- DateTime: instancia de TLuaDateTime que se obtiene de otra función de clase o se construye.

Establecer

Set(int iNewTime)

Parámetros

- iNew: la compensación de tiempo en segundos o el tiempo absoluto en segundos desde el 1 de enero de 1970.

Comentarios

Esta función se puede usar para crear una instancia de TLuaDateTime que después se suma a otra instancia de TLuaDateTime, se resta de ella o se compara con ella.

Sub

Sub(TLuaDateTime DateTime)

Parámetros

- DateTime: instancia de TLuaDateTime que se obtiene de otra función de clase o se construye.

Comentarios

La función resta la hora contenida en el parámetro DateTime de la hora almacenada en el activo.

Capítulo 6

TLuaDB

Esta clase proporciona funciones para consultar bases de datos SQL.

En este capítulo

Script de ejemplo: TLuaDB

```
--Create new DB asset
DB = TLuaDB();

--Connect to a DSN
if (DB:Connect("DSN=testdsn;",TLuaDB.CLIENT_ODBC) == true) then

  --Insert a few rows
  bok = DB:Execute("insert into test (iID,sTest) values(10,'test');");

  --Select all rows in table
  bok = DB:Execute("select * from test;");
  if ( bok == true) then
        --Check if we got rows back
        if(DB:ResultAvilable() == true) then
              --Print how many columns this table contains
              iColCount = DB:ColCount(); print("Columns in this table:
"..iColCount);
              --Get first row
              while (DB:NextRow() == true) do
                    for iCurrentCol = 1, iColCount do
                          --GetColType and GetCol take a 1 based index
                          iColType = DB:GetColType(iCurrentCol);
                          sData = DB:GetCol(iCurrentCol);
                          --Print column #, column type and data
                          print("Col #"..iCurrentCol.." Type: "..
iColType.." Data: "..sData);
                    end
              end
        end
  else
        --Print error and exit
        SetExitStatus("Failed" .. DB:GetErrorDescription(),false);
  end
else
  --Print error and exit
  SetExitStatus("Failed to connect"..DB:GetErrorDescription(),false);
end
```

ColCount

int ColCount()

Valores de devolución

Devuelve la cantidad de columnas en el resultado de una consulta realizada correctamente.

Conectar

bool Connect(string sConnectString,int iClientType=TLuaDB.CLIENT_ODBC)

Valores de devolución

El resultado es verdadero si la conexión se ejecutó correctamente; si ocurrió un error, el resultado es falso.

Parámetros

- sConnectString: una cadena de conexión exclusiva del cliente. Consulte la sección de comentarios para obtener más información.
- iClientType: tipo de cliente con el que comunicarse. Consulte las opciones a continuación.

Comentarios

La cadena de conexión es exclusiva del cliente. A continuación, encontrará los clientes compatibles actualmente.

CLIENT _ODBC

Ejemplo de cadena de conexión:

```
sConnectString = "DSN=test;UID=test;PWD=test";
```

Esta cadena de conexión utiliza un origen de datos con el nombre “test” y proporciona el nombre de usuario (UID) “test” y la contraseña (PWD) “test” al DSN.

Si no se necesita un nombre de usuario o una contraseña, la conexión puede tener el siguiente formato:

```
sConnectString = "DSN=test;";
```

Network Monitor se ejecuta como un servicio. El origen de datos debe ser un origen de datos de sistema. Esto es distinto del IDE que también puede usar un usuario DSN.

CLIENT _SQLSERVER

Ejemplo de cadena de conexión:

```
sConnectString = "myserver@mydatabase";
```

Para conectar una instancia con nombre de SQL Server 2000, use lo siguiente:

```
sConnectString = "myserver\\instance_name@mydatabase";
```

CLIENT_MYSQL

Ejemplo de cadena de conexión (con el uso del puerto 3306 predeterminado):

```
sConnectString =  myserver@mydatabase
```

Conexión a un servidor que escucha un puerto predeterminado

```
sConnectString = "myserver:portnumber@mydatabase";
```

Tenga en cuenta que se debe instalar la biblioteca de cliente MySQL —“MySQL Workbench” o “Connector/C (libmysql)”— y se la debe incluir en la variable PATH predeterminada del sistema Windows para que **Network Monitor** y el IDE la puedan encontrar.

Connect(2)

bool Connect(string sConnectString,string sUser,string sPassword,int iClientType=TLuaDB.CLIENT_ODBC)

Valores de devolución

El resultado es verdadero si la conexión se ejecutó correctamente; si ocurrió un error, el resultado es falso.

Parámetros

- sConnectString: una cadena de conexión exclusiva del cliente. Consulte la sección de comentarios para obtener más información.
- sUsername: credenciales para usar con la conexión.
- sPassword: credenciales para usar con la conexión. No puede estar vacío si se especifica un nombre de usuario.
- iClientType: tipo de cliente con el que comunicarse. Actualmente, la única opción es CLIENT_ODBC.

Comentarios

La cadena de conexión es exclusiva del cliente. Consulte **Conectar** *(página 30)* para obtener más información.

Ejecutar

bool Execute(string sSQL)

Valores de devolución

El resultado es verdadero si la consulta se ejecutó correctamente; si ocurrió un error, el resultado es falso.

Parámetros

- sSQL: una instrucción SQL.

Comentarios

Si ocurre un error al ejecutar una instrucción SQL, la función GetErrorDescription() devuelve una cadena con una descripción del error.

GetCol

string GetCol(int iIndex)

Valores de devolución

Devuelve una cadena con los datos recuperados de la columna.

Parámetros

- iIndex: un índice de columna de base 1.

Comentarios

Tenga en cuenta que el índice de columna es de base 1. Si ColCount() devuelve el número 10, el intervalo de índices válido es 1-10. Para recuperar el tipo de datos, invoque la función GetColType().

GetCol_AsDateTime

TLuaDateTime GetCol_AsDateTime(int iIndex)

Valores de devolución

Devuelve una estructura TLuaDateTime con la fecha y la hora recuperadas de la columna.

Parámetros

- iIndex: un índice de columna de base 1.

Comentarios

Tenga en cuenta que el índice de columna es de base 1. Si ColCount() devuelve el número 10, el intervalo de índices válido es 1-10. Esta función no se debe invocar si la columna no es de tipo de fecha y hora.

GetColType

int GetColType(int iIndex)

Valores de devolución

Devuelve un entero que representa el tipo de datos que contiene la columna.

Parámetros

- iIndex: un índice de columna de base 1.

Comentarios

La función GetCol() siempre devuelve los datos en forma de cadena. La función GetColType determina el tipo de datos en que se puede convertir la cadena después de la extracción. Existen los siguientes tipos:

- TYPE_BOOL: valor booleano
- TYPE_NUMERIC: numérico
- TYPE_SHORT: breve
- TYPE_LONG: extenso
- TYPE_DOUBLE: doble (real)
- TYPE_DATETIME: fecha y hora
- TYPE_STRING: cadena
- TYPE_UNKNOWN: tipo de campo desconocido/incompatible
- TYPE_BYTES: bytes
- TYPE_LONG_BINARY: binario extenso
- TYPE_LONG_CHAR: caracteres extenso
- TYPE_BLOB: dispositivo binario
- TYPE_DBMS_SPECIFIC: datos exclusivos del cliente (sin conversión)

GetErrorDescription

string GetErrorDescription(void)

Valores de devolución

Devuelve la descripción del error más reciente generada por la API de TLuaDB.

NextRow

bool NextRow()

Valores de devolución

El resultado es verdadero si se obtiene un nuevo resultado. Si no existen más resultados para la consulta, el resultado es falso.

Comentarios

La función recupera un nuevo resultado generado por una invocación previa de la función Execute. Esta función se debe invocar antes de la primera invocación de las funciones ColCount, GetCol o GetColType.

ResultAvilable

bool ResultAvilable()

Valores de devolución

El valor devuelto es verdadero si una instrucción SQL ejecutada devuelve cualquier resultado.

Comentarios

Tal como fue diseñada, no inserta, actualiza ni elimina instrucciones que devuelven cualquier resultado después de la ejecución. Antes de invocar esta función, se debe usar el valor de devolución de la función Execute para determinar si se ejecutó correctamente una instrucción.

Capítulo 7

TLuaDNS

Esta clase proporciona funciones para consultar servidores DNS.

En este capítulo

Script de ejemplo: TLuaDNS

```
DNS = TLuaDNS();
DNS:Begin(true);

if DNS:Query("microsoft.com",TLuaDNS.LuaDNS_TYPE_MX,false) then

  Record = TLuaDNS_MXRecord();

  while (DNS:Next(Record)) do
        print(Record.m_sNameExchange);
  end

  SetExitStatus("Test ok",true);

else
  SetExitStatus("Test failed",false);
end
```

Comenzar

bool Begin(bool bUselocalhost=false)

Valores de devolución

Valor distinto de cero si la función se ejecutó correctamente; de lo contrario, devuelve cero.

Parámetros

- bUselocahost: se establece en verdadero si se desea hacer una consulta al servidor DNS en el equipo host de **Network Monitor**.

Comentarios

La función inicia una consulta DNS. Invocar la función Query antes de que se ejecute Begin puede dar resultados desconocidos.

Final

void End()

Comentarios

Esta función finaliza una transacción y restablece el activo para que se pueda ejecutar una nueva consulta. Si no se invoca esta función, el resultado de la siguiente consulta es impredecible.

GetErrorDescription

string GetErrorDescription(void)

Valores de devolución

Devuelve la descripción del error más reciente generada por la API de TLuaDNS.

Siguiente

bool Next(TLuaDNS_NSRecord Record);
bool Next(TLuaDNS_CNAMERecord Record);
bool Next(TLuaDNS_ARecord Record);
bool Next(TLuaDNS_PTRRecord Record);
bool Next(TLuaDNS_SOARecord Record);
bool Next(TLuaDNS_MXRecord Record);
bool Next(TLuaDNS_TXTRecord Record);

Valores de devolución

Valor distinto de cero si la función se ejecutó correctamente; de lo contrario, devuelve cero.

Parámetros

Registro de tipo DA DNS que recibe la información sobre la que se consultó al servidor DNS.

Comentarios

La función devuelve un resultado verdadero si recuperó correctamente un nuevo registro; si la función devuelve un resultado falso, no hay más registros para recuperar.

Query

bool Query(string sDomainName,int iRecordType,bool bBypassCache=false);

Valores de devolución

Valor distinto de cero si la función se ejecutó correctamente; de lo contrario, devuelve cero.

Parámetros

- sDomainName: el dominio para consultar.
- iRecordType: uno de los tipos de registros incluidos en la sección de comentarios.
- bBypassCache: es falso como opción predeterminada. Si se establece en verdadero, la consulta omite la resolución local y realiza la consulta directamente al servidor DNS.

Comentarios

La función envía una consulta al servidor DNS. Los resultados se pueden extraer con una invocación o más de la función Next().

Se pueden invocar los siguientes tipos de registros:

- LuaDNS_TYPE_PTR
- LuaDNS_TYPE_TEXT
- LuaDNS_TYPE_SOA
- LuaDNS_TYPE_CNAME
- LuaDNS_TYPE_MX
- LuaDNS_TYPE_NS
- LuaDNS_TYPE_A

Puede encontrar referencias sobre tipos de registros DNS en la siguiente dirección:
http://www.iana.org/assignments/dns-parameters *(http://www.iana.org/assignments/dns-parameters)*

TLuaDNS_ARecord

Miembros de la clase

- int m_iTTL
- string m_sIPAddress;

TLuaDNS_CNAMERecord

Miembros de la clase

- int m_iTTL
- string m_sHostname

TLuaDNS_MXRecord

Miembros de la clase

- int m_iPreference
- string m_sNameExchange

TLuaDNS_NSRecord

Miembros de la clase

- int m_iTTL
- string m_sHostname

TLuaDNS_PTRRecord

Miembros de la clase

- int m_iTTL
- string m_sHostname

TLuaDNS_SOARecord

Miembros de la clase

- int m_iTTL
- string m_sPrimaryServer
- string m_sAdministrator
- int m_iSerialNo
- int m_iRefresh
- int m_iRetry
- int m_iExpire
- int m_iDefaultTTL

TLuaDNS_TXTRecord

Miembros de la clase

- int StringCount(void)
- string GetString(int iPos)

Capítulo 8

TLuaFile

Esta clase proporciona rutinas básicas de manejo de archivos y es compatible con archivos binarios y de texto. Todas las operaciones de archivos están relacionadas con el contexto en el que se ejecuta el script. Por ejemplo, si un activo ejecuta el script con la dirección `\\myserver`, la ruta de acceso del archivo, `c:\test.txt`, se traduce como `\\myserver\C$\test.txt`.

Existe sólo una excepción, y se da cuando la clase se inicializa con el valor verdadero. En ese caso, todas las operaciones se relacionan con el equipo host de **Network Monitor**. Para obtener más información, consulte el script de ejemplo no. 3 en esta clase.

En este capítulo

Scripts de muestra: TLuaFile

Ejemplo 1

```
--Script creates a new text file, writes a string to it and close it.
file = TLuaFile()

--Open a text file
iRet = file:Open("c:\\test.txt",true)

--Check if it could be created, it might exist and be write protected
if iRet==0 then
  sErrString = "Failed to create file, error code:"..GetLastError().."\n"
  SetExitStatus(sErrString,false)
else
  print("File created\n")
  --Write a string to the file
  sString = "Hello world!" file:Write(sString,string.len(sString))
  --Close the file
  file:Close() SetExitStatus("Test ok",true)
end
```

Ejemplo 2

```
--Script demonstrates:
--File enumration
--Directory enumration
--Create and delete a directory

--Construct a new file device
file = TLuaFile();

--Scan the directory c:\temp for file using the wildcard *.*
sResult = file:GetFileList("c:\\temp","*.*")
print(sResult)

--Scans the directory c:\temp for sub directories
sResult = file:GetDirectoryList("c:\\temp")
print(sResult)
bResult = false

--Create a directory called "temp20" on the c: harddisk
if file:CreateDirectory("c:\\temp20") then
  print("Created directory");
  bResult = true
else
  print("Failed to create directory")
end

--Delete the directory we created above
if file:DeleteDirectory("c:\\temp20") then
  bResult = true
  print("Deleted directory");
else
  print("Failed to delete directory")
end
```

```
if bResult then
  SetExitStatus("Test ok",true)
else
  SetExitStatus("Test failed",false
end
```

Ejemplo 3

```
--KNM Lua API example (C) 2006 Kaseya AB
--Script creates a new text file, writes a string to it and close it.

--Switch the context so that files are opened on the KNM host machine,
not translating the paths
file = TLuaFile(true)

--Open a text file
iRet = file:Open("c:\\test.txt",true)

--Check if it could be created, it might exist and be write protected
if iRet==0 then
  sErrString = "Failed to create file, error code:"..GetLastError().."\n"
  SetExitStatus(sErrString,false)
else
  print("File created\n")
  --Write a string to the file
  sString = "Hello world!"
  file:Write(sString,string.len(sString))
  --Close the file
  file:Close()
  SetExitStatus("Test ok",true)
end
```

Cerrar

int Close()

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Comentarios

Cierra el archivo y lo pone como no disponible para lectura y escritura. Si no cerró el archivo antes de que se destruya el activo, el destructor lo hará por usted.

CopyFile

int CopyFile(string sSource,string sDest)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- sSource: ruta de acceso y nombre del archivo por copiar.
- sDest: nueva ruta de acceso y nombre del archivo nuevo.

Comentarios

La función copia un archivo. No se pueden copiar directorios con esta función.

CreateDirectory

int CreateDirectory(string sPath)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- sPath: ruta de acceso del directorio por crear.

Comentarios

Debe existir el directorio primario del directorio por crear. De lo contrario, se produce un error en la función.

DeleteDirectory

int DeleteDirectory(string sDirectory)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- sDirectory: ruta de acceso del directorio por borrar.

Comentarios

El directorio no puede estar vacío y no puede ser un directorio raíz o el directorio de trabajo actual.

DeleteFile

int DeleteFile(string sFileName)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- sFileName: la ruta de acceso y el nombre del archivo por eliminar.

Comentarios

La función elimina un archivo. No se pueden eliminar directorios con esta función.

DoesFileExist

int DoesFileExist(string sFile);

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- sFile: la ruta de acceso y el nombre del archivo.

Comentarios

Si existe el archivo, devuelve un valor distinto de cero.

GetDirectoryList

string GetDirectoryList(string sDirectory)

Valores de devolución

Devuelve una cadena con los subdirectorios separados por un carácter de retorno de carro; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- sDirectory: la ruta de acceso del directorio por buscar.

Comentarios

La cadena de retorno contiene todos los subdirectorios en el directorio especificado. Cada directorio se devuelve con su ruta de acceso completa. Los directorios de la cadena están separados por un carácter de retorno de carro.

GetFileAccessedTime

TLuaDateTime GetFileAccessedTime(string sFilename)

Valores de devolución

Devuelve un activo TLuaDateTime que contiene la hora en que se accedió al archivo por última vez. De lo contrario, devuelve un activo TLuaDateTime que contiene el valor 0. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- sFilename: la ruta de acceso y el nombre del archivo.

Comentarios

Usa el activo TLuaDateTime para construir una cadena que representa el tiempo almacenado en el activo o lo compara con otro activo TLuaDateTime.

GetFileCreatedTime

TLuaDateTime GetFileCreatedTime(string sFilename)

Valores de devolución

Devuelve un activo TLuaDateTime que contiene la hora en que se creó el archivo. De lo contrario, devuelve un activo TLuaDateTime que contiene el valor 0. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- pFilename: la ruta de acceso y el nombre del archivo.

Comentarios

Usa el activo TLuaDateTime para construir una cadena que representa el tiempo almacenado en el activo o lo compara con otro activo TLuaDateTime.

GetFileList

string GetFileList(string sDirectory,string sWildCard)

Valores de devolución

Devuelve una cadena con los archivos enumerados separados por un carácter de retorno de carro. De lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- sDirectory: ruta de acceso del directorio.
- sWildCard: el comodín para filtrar los archivos deseados. Para recuperar todos los archivos del directorio, use el comodín *.*.

Comentarios

La cadena de devolución contiene todos los archivos del directorio que coinciden con el comodín proporcionado. Cada archivo se devuelve con su ruta de acceso completa. Los archivos de la cadena están separados por un carácter de retorno de carro.

GetFileModifiedTime

TLuaDateTime GetFileModifiedTime(string sFilename)

Valores de devolución

Devuelve un activo TLuaDateTime que contiene la hora en que se modificó el archivo por última vez. De lo contrario, devuelve un activo TLuaDateTime que contiene el valor 0. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- sFilename: la ruta de acceso y el nombre del archivo.

Comentarios

Usa el activo TLuaDateTime para construir una cadena que representa el tiempo almacenado en el activo o lo compara con otro activo TLuaDateTime.

GetFileSize

int GetFileSize(string sFile)

Valores de devolución

Se devuelve el tamaño del archivo en cantidad de bytes si la función se ejecuta correctamente. Si no se puede acceder al archivo, el valor de devolución es -1.

Parámetros

- sFile: la ruta de acceso y el nombre del archivo.

Comentarios

La función se limita a archivos con menos de 2^31 bytes.

GetFileSizeMB

int GetFileSizeMB(string sFile)

Valores de devolución

Se devuelve el tamaño del archivo en cantidad de megabytes si la función se ejecuta correctamente. Si no se puede acceder al archivo, el valor de devolución es -1.

Parámetros

- sFile: la ruta de acceso y el nombre del archivo.

GetFileStatus

int GetFileStatus(string sFile)

Valores de devolución

Se devuelve un valor que describe el estado actual del archivo si la función se ejecuta correctamente. Si no se puede acceder al archivo, el valor de devolución es -1.

Parámetros

- sFile: la ruta de acceso y el nombre del archivo.

Comentarios

La función devuelve una combinación de los siguientes indicadores para describir el estado del archivo.

FILE_NORMAL = 0x00	Archivo normal.
FILE_READONLY = 0x01	El archivo es de sólo lectura.
FILE_HIDDEN = 0x02	El archivo está oculto.
FILE_SYSTEM = 0x04	El archivo es un archivo de sistema.
FILE_VOLUME = 0x08	El archivo es un volumen.
FILE_DIR = 0x10	El archivo es un directorio.
FILE_ARCHIV = 0x20	El archivo tiene establecido el bit de archivo.

MoveFile

int MoveFile(string sSource,string sDest)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- sSource: la ruta de acceso y el nombre del archivo por mover. sDest: la nueva ruta de acceso y el nuevo nombre del archivo.

Comentarios

La función mueve un archivo. Debe existir el directorio al que se moverá el archivo. De lo contrario, se produce un error en la función. No se pueden mover directorios con esta función.

Abra

int Open(string sFileName, bool bCreate=false)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- sFileName: el nombre y la ruta de acceso del archivo por abrir o crear. bCreate: si está establecido en un valor distinto de cero, la función crea un nuevo archivo. Si ya existe el archivo especificado, su contenido se destruye.

Comentarios

Si bCreate está establecido en cero y el archivo no existe, se produce un error en la función.

Leer

string,int Read(int iSize)

Valores de devolución

Si la función se ejecutó correctamente e iSize está establecido en el tamaño de los datos devueltos, se devuelve una cadena. De lo contrario, no devuelve nada. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- iSize: el tamaño de los datos por leer, en bytes.

Comentarios

Esta función lee el tamaño especificado de los datos en el archivo y mueve el puntero a archivo hacia delante la misma cantidad. Si se llega al final del archivo, se frena la lectura y se devuelven los datos que se leyeron hasta que se llegó al final del archivo.

ReadData

local data,int ReadData(int iSize)

Valores de devolución

La función devuelve un tipo de datos especial que sólo se puede utilizar junto con ConvertFromUTF16. Si la función se ejecutó correctamente, iSize se establece en el tamaño de los datos devueltos. De lo contrario, no devuelve nada. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- iSize: el tamaño de los datos por leer, en bytes.

Comentarios

Esta función lee el tamaño especificado de los datos en el archivo y mueve el puntero a archivo hacia delante la misma cantidad. Si se llega al final del archivo, se frena la lectura y se devuelven los datos que se leyeron hasta que se llegó al final del archivo.

RenameFile

int RenameFile(string sOrgFile,string sNewFile)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- sOrgFile: la ruta de acceso y el nombre del archivo al que se le desea cambiar el nombre.
- sNewFile: la nueva ruta de acceso y el nuevo nombre del archivo.

Comentarios

La función cambia el nombre de un archivo. No se pueden cambiar nombres de directorios con esta función.

SeekFromCurrent

int SeekFromCurrent(int iNumberofBytes)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- iNumberofBytes: la cantidad de bytes, relativa a la posición actual, en la que se debe reposicionar el puntero a archivo.

Comentarios

La función mueve el puntero a archivo en relación con la posición actual. Se pueden especificar tanto valores negativos como positivos. El puntero se puede posicionar después del final del archivo; esto borra el marcador del final del archivo.

SeekFromEnd

int SeekFromEnd(int iNumberofBytes)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- iNumberofBytes: la cantidad de bytes, contada desde el final del archivo, en la que se debe reposicionar el puntero a archivo.

Comentarios

La función mueve la posición actual del puntero a archivo el valor especificado en iNumberofBytes desde el final del archivo. Tenga en cuenta que iNumberofBytes debe ser negativo para poder mover el puntero a archivo “hacia arriba” en el archivo.

SeekFromStart

int SeekFromStart(int iNumberofBytes)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- iNumberofBytes: la cantidad de bytes, contada desde el comienzo del archivo, en la que se debe reposicionar el puntero a archivo.

Comentarios

La función mueve la posición actual del puntero a archivo el valor especificado en iNumberofBytes desde el comienzo del archivo. El puntero se puede posicionar después del final del archivo; esto borra el marcador del final del archivo.

Escribir

int Write(string sData,int iSize)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- sData: la matriz de datos por escribir en el archivo.
- iSize: el tamaño de los datos por escribir.

Comentarios

La función escribe desde la posición actual del puntero a archivo. Si es necesario, extiende el archivo cuando pasa el final del archivo actual. Si el archivo abierto está protegido contra escritura, se produce un error en la función.

Capítulo 9

TLuaFTPClient

Esta clase implementa un cliente FTP con capacidad para operaciones FTP básicas.

En este capítulo

Script de ejemplo: TLuaFTPClient

```
--Script connects to a FTP server and download content of file
ftp = TLuaFTPClient();

--Enter the username and password for the session
sUsername = "myusername" sPassword = "mypassword"

--Connect to FTP server using username and password
iRet = ftp:Connect(sUsername,sPassword,21)

--Check return value from server
if iRet == 0 then
  --Failed to connect, print why
  iRet = GetLastError()
  sErrorString = FormatErrorString(iRet)
  sErrString = "Error when connecting to FTP server, error: "..sErrorString
  SetExitStatus(sErrString,false)

else
  --Open a file on the FTP server that we know exist
  sFilename = "update.vcf"

  --Open file, do not create it, use text mode
  iRet = ftp:OpenFile(sFilename,false,true)
  if iRet == 0 then
        sErrString = "Cannot open file "..sFilename
SetExitStatus(sErrString,false)
  else
        iMaxSize = 1024*16
        --Read a number of bytes from the file
        --Note here that we are using the special lua return value convention
        --Read returns one string and the size of the string
        sFilecontent, iMaxSize = ftp:Read(iMaxSize)
        print("Size of content: "..iMaxSize.."\n")
        print(sFilecontent)
        --Close file so we can open a new file later or close the session
        ftp:CloseFile()
        SetExitStatus("Test ok",true)
  end
end

--Close FTP session
ftp:Close()
```

ChangeDirectory

int ChangeDirectory(string sDir)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- sDir: la ruta de acceso del nuevo directorio actual.

Comentarios

Si no existe el directorio, se produce un error en la función.

Cerrar

Close()

Comentarios

La función cierra la conexión actual al servidor FTP. Se debe invocar la función para cerrar la conexión actual.

CloseFile

void CloseFile()

Comentarios

Cierra un archivo abierto con OpenFile.

Conectar

bool Connect(unsigned int iPort=21)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- sUsername: cadena que contiene el nombre de usuario.
- sPassword: cadena que contiene la contraseña.
- iPort: puerto predeterminado 21 (puerto FTP estándar).

Comentarios

La función establece una conexión a un servidor FTP con el nombre de usuario y la contraseña proporcionados. Si el servidor FTP está asociado a otro puerto, se puede modificar el puerto predeterminado 21.

CreateDirectory

int CreateDirectory(string sDir)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- sDir: la ruta de acceso del directorio por crear.

Comentarios

Debe existir el directorio donde se creará el nuevo directorio. De lo contrario, se produce un error en la función.

DeleteDirectory

int DeleteDirectory(string sDir)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- sDir: la ruta de acceso del directorio por eliminar.

Comentarios

Si el directorio no está vacío, se produce un error en la función.

DeleteFile

int DeleteFile(string sFilename)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- sFilename: la ruta de acceso y el nombre del archivo por eliminar.

Comentarios

Si no existe el archivo, se produce un error en la función.

FindDirectory

string FindDirectory()

Valores de devolución

Devuelve una cadena con los directorios enumerados separados por un carácter de retorno de carro; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Comentarios

La cadena de retorno contiene todos los subdirectorios del directorio actual. Cada directorio se devuelve con su ruta de acceso completa. Los directorios de la cadena están separados por un carácter de retorno de carro.

FindFile

string FindFile(string sWildcard)

Valores de devolución

Devuelve una cadena con los archivos enumerados separados por un carácter de retorno de carro. De lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- sWildCard: el comodín para filtrar los archivos deseados. Para recuperar todos los archivos del directorio, use el comodín *.*.

Comentarios

La cadena de devolución contiene todos los archivos del directorio actual que coinciden con el comodín proporcionado. Cada archivo se devuelve con su ruta de acceso completa. Los archivos de la cadena están separados por un carácter de retorno de carro.

GetCurrentDirectory

int GetCurrentDirectory(string sDir)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- sDir: la ruta de acceso del nuevo directorio actual.

Comentarios

Si no existe el directorio, se produce un error en la función.

GetFileModifiedTime

TLuaDateTime GetFileModifiedTime(string sFilename)

Valores de devolución

Si la función se ejecutó correctamente, se devuelve un activo TLuaDateTime que contiene la hora de la última modificación del archivo. De lo contrario, se devuelve un activo TLuaDateTime establecido en 0.

Parámetros

- sFilename: el nombre del archivo en el directorio actual.

Comentarios

El archivo debe estar en el directorio actual para que esta función se ejecute correctamente. Las rutas de acceso relativas no funcionan.

GetFileSize

int GetFileSize(string sFilename)

Valores de devolución

Se devuelve el tamaño del archivo en cantidad de bytes si la función se ejecuta correctamente. Si no se puede acceder al archivo, el valor de devolución es -1.

Parámetros

- sFilename: la ruta de acceso y el nombre del archivo.

Comentarios

La función se limita a archivos con menos de 2^31 bytes.

OpenFile

int OpenFile(string sFilename,bool bWrite,bool bText)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- sFilename: el nombre del archivo que se desea abrir.
- bWrite: si es un valor distinto de cero, el archivo se abre en modo de escritura. Si el valor es cero, el archivo se abre en modo de sólo lectura.
- bText: si es un valor distinto de cero, el archivo se abre en modo de traducción de texto. De lo contrario, se abre en modo binario.

Comentarios

La función abre un archivo en el servidor FTP. Después de que se abre el archivo, se puede invocar la función de lectura y escritura. Para cerrar el archivo, se usa la función CloseFile.

Leer

string Read(int iSize)

Valores de devolución

Se devuelve una matriz con los datos leídos en el archivo. De lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- iSize: el tamaño de los datos por leer en el archivo. Cuando se devuelve la función, contiene información sobre cuánto se leyó, que puede ser una cantidad inferior al tamaño requerido.

Comentarios

Si no se abrió un archivo, se produce un error en la función.

RenameFile

bool RenameFile(string sOldname,string sNewname)

Valores de devolución

Si la función se ejecutó correctamente, el resultado es verdadero. De lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- sOldname: el nombre antiguo del archivo al que se le cambiará el nombre. sNewname: el nuevo nombre del archivo.

Comentarios

Si existe un archivo con el mismo nombre que se proporcionó en el segundo parámetro, se produce un error en la función.

Escribir

int Write(string sData,int iSize)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- sData: la matriz de datos por escribir en el archivo.
- iSize: el tamaño de la matriz por escribir.

Comentarios

Si no se abrió un archivo o si el archivo está abierto en modo de sólo lectura, se produce un error en la función. También se puede producir un error en la función si el espacio de almacenamiento en el servidor FTP está agotado.

Capítulo 10

TLuaHTTPClient

Esta clase implementa un cliente HTTP básico.

En este capítulo

Script de ejemplo: TLuaHTTPClient

```
--Script to download a html page from a web server
http = TLuaHTTPClient()

--Connect using the default parameters
iRet = http:Connect()
if iRet ~= 0 then

  --Make a GET request to default document
  iRet = http:Get("/")

  --Print returned code from HTTP server
  print("Code:"..iRet)

  --Extract content length
  iRet = http:GetHeaderContentLength()
  print("Content length:"..iRet)

  --Print content
  string,iRet = http:GetContent(iRet)
  print(string)

  --Print raw headers
  string = http:GetHeadersRaw()
  print("headers:\n"..string)

  --Print cookies
  string = http:GetHeaderCookies()
  print("Cookies:\n"..string)

  --Extract and print cookies one by one
  iNumber = http:GetHeaderCookieCount()
  for count = 0, iNumber-1 do
        string = http:GetHeaderCookie(count)
        print("Cookie #"..count.." "..string.."\n")
  end

  --Extract location header
  string = http:GetHeaderLocation();
  print("location:\n"..string)
  SetExitStatus("Test ok",true)

else

  print("Connect failed")
  SetExitStatus("Test failed",false)

end
```

Conectar

bool Connect(bool bSecure,unsigned short iPort)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- iPort: el número de puerto al cuál conectarse. El puerto predeterminado es 80.
- bSecure: si la conexión se establecerá mediante HTTPS, está establecido en un valor distinto de cero.
- sUsername: nombre de usuario optativo para los servidores que requieren autenticación.
- sPassword: contraseña optativa para los servidores que requieren autenticación.

Comentarios

La función establece una conexión a un servidor HTTP con los parámetros proporcionados. Se debe invocar esta función antes de invocar cualquier otra función en esta clase.

Cerrar

Close()

Comentarios

Cierra una conexión abierta.

Get

int Get(string sUrl,string sHeaders=NULL)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- sUrl: URL relativa a la URL base del sitio.
- sHeaders: cadena de encabezado optativa que contiene los encabezados que se deben enviar con la solicitud.

Comentarios

La conexión siempre está abierta en el contexto del activo. Por lo tanto, la URL proporcionada a la función siempre debe ser relativa a la URL base.

Post

int Post(string sUrl,string sHeaders=NULL,string sData=NULL)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- sUrl: URL relativa a la URL base del sitio.
- sHeaders: cadena de encabezado optativa que contiene los encabezados que se deben enviar con la solicitud.
- sData: datos optativos para incluir en la solicitud post.

Comentarios

La conexión siempre está abierta en el contexto del activo. Por lo tanto, la URL proporcionada a la función siempre debe ser relativa a la URL base. Todos los encabezados proporcionados deben terminar con un par CR/LF.

GetContent

string GetContent(int iSize)

Valor de devolución

Si se ejecuta correctamente, devuelve una cadena con la porción de contenido de una solicitud GET; de lo contrario, no devuelve nada. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- iSize: Cuando se devuelve la función, se establece en el tamaño de la cadena devuelta.

Comentarios

El contenido hace referencia a los datos que devuelve una solicitud que sigue al encabezado.

GetHeadersRaw

string GetHeadersRaw()

Valores de devolución

Si se ejecuta correctamente, devuelve una cadena que contiene los encabezados devueltos por la solicitud. De lo contrario, se devuelve una cadena vacía.

Comentarios

Los encabezados se devuelven exactamente como los envía el servidor.

GetHeaderLocation

string GetHeaderLocation()

Valores de devolución

Si se ejecutó correctamente, devuelve una cadena con el encabezado “Location:”. De lo contrario, se devuelve una cadena vacía.

GetHeaderContentLength

int GetHeaderContentLength()

Valores de devolución

La longitud en bytes de la porción de contenido de la solicitud.

GetHeaderContentType

string GetHeaderContentType()

Valor de devolución

Si se ejecutó correctamente, devuelve una cadena con el encabezado “Content-Type:”. De lo contrario, se devuelve una cadena vacía.

GetHeaderContentTransferEncoding

string GetHeaderContentTransferEncoding()

Valor de devolución

Si se ejecutó correctamente, devuelve una cadena con el encabezado “Transfer-Encoding:”. De lo contrario, se devuelve una cadena vacía.

GetHeaderCookies

string GetHeaderCookies()

Valor de devolución

Si se ejecuta correctamente, devuelve una cadena que contiene todas las cookies devueltas por el servidor separadas por un carácter de retorno de carro. De lo contrario, devuelve una cadena vacía.

GetHeaderCookie

string GetHeaderCookie(int iIndex)

Valor de devolución

Se devuelve una cadena con la cadena de cookies solicitada.

Parámetros

- iIndex: un índice de base cero que especifica la cookie por devolver.

Comentarios

Si se especifica un número negativo o un índice fuera de intervalo, se devuelve una cadena vacía.

GetHeaderCookieCount

int GetHeaderCookieCount()

Valor de devolución

Se devuelve la cantidad de cookies devueltas por la solicitud.

GetHeaderDate

string GetHeaderDate()

Valor de devolución

Si se ejecutó correctamente, se devuelve una cadena con el encabezado “Date:”. De lo contrario, se devuelve una cadena vacía.

GetHeaderExpires

string GetHeaderExpires()

Valor de devolución

Si se ejecutó correctamente, se devuelve una cadena con el encabezado “Expires:”. De lo contrario, se devuelve una cadena vacía.

GetHeaderHost

string GetHeaderHost()

Valor de devolución

Si se ejecutó correctamente, se devuelve una cadena con el encabezado “Host:”. De lo contrario, se devuelve una cadena vacía.

Capítulo 11

TLuaICMP

Esta clase proporciona funciones de ping y de seguimiento de ruta que se pueden utilizar para diagnosticar una conexión de red.

En este capítulo

Script de ejemplo: TLuaICMP

```
--Description: Trace route example
icmp = TLuaICMP()

iPacketSize = 32 --packet size in bytes, excluding the header
bNoFragment = false --Set to true to inhibit fragmentation of packet sent
iMaxHops = 255  --Max number of hops in route

--Begin trace
bok = icmp:BeginTrace(iPacketSize,bNoFragment,iMaxHops)
if bok ~= true then
  SetExitStatus("Trace failed!",false);
end

--Print the result
iCount = 1;
Result = TLuaICMPTraceResult()
while icmp:NextTraceResult(Result) do
  print("Hop: "..iCount)
  print(Result.m_Name)
  print(Result.m_iRoundTripTimeMs)
  iCount = iCount + 1
end

--Clean up resources
icmp:EndTrace()
SetExitStatus("Trace ok!",true);
```

BeginTrace

bool BeginTrace(int iPacketsToSend,int iPacketSize,bool bNoFragment,int iTimeoutMs)

Valores de devolución

Esta función devuelve un resultado verdadero si el seguimiento fue satisfactorio y un resultado falso si se produjo un error.

Parámetros

- iPacketsToSend: un entero positivo entre 1 y 255.
- iPacketSize: el tamaño de los paquetes por enviar; un entero entre 0 y 65500.
- bNoFragment: se establece en verdadero para evitar que los paquetes enviados se fragmenten. Se produce un error en la función y se devuelve un resultado falso si la opción está establecida y se fragmenta el paquete.
- iTimeoutMs: el tiempo máximo en milisegundos que la función espera para que se devuelva el paquete.

Comentarios

Si la función bNoFragment se establece en verdadero, es posible probar el máximo tamaño de trama para una ruta. Ajuste iPacketSize hasta que se produzca un error en la función debido a la fragmentación.

EndTrace

EndTrace()

Comentarios

Esta función realiza una limpieza de los recursos utilizados. Se debe invocar para cada invocación de BeginTrace().

NextTraceResult

bool NextTraceResult(TLuaICMPTraceResult Result)

Valores de devolución

La función devuelve un resultado verdadero mientras haya un resultado disponible.

Parámetros

- Resultado: una variable **TLuaICMPTraceResult** que recibe el resultado para el salto actual.

Comentarios

Para iterar en el conjunto de resultados, invoque la función hasta que se devuelva un resultado nulo.

Ping

bool Ping(TLuaICMPPingResult Result,int iPacketsToSend,int iPacketSize,bool bNoFragment,int iTimeoutMs)

Valores de devolución

La función devuelve un activo TLuaICMPPingResult que contiene el resultado de la operación.

Parámetros

- iPacketsToSend: un entero positivo entre 1 y 255.
- iPacketSize: el tamaño de los paquetes por enviar; un entero entre 0 y 65500.
- bNoFragment: se establece en verdadero para evitar que los paquetes enviados se fragmenten. Se produce un error en la función y se devuelve un resultado falso si la opción está establecida y se fragmenta el paquete.
- iTimeoutMs: el tiempo máximo en milisegundos que la función espera para que se devuelva el paquete.

Comentarios

Si establece el argumento bNoFragment, se puede probar el máximo tamaño de trama posible para una ruta.

Capítulo 12

TLuaICMPPingResult

TLuaICMPPingResult es una clase de sólo lectura.

Miembros de la clase

- int m_iRoundTripTimeMs
- float m_fPacketloss

Capítulo 13

TLuaICMPTraceResult

TLuaICMPTraceResult es una clase de sólo lectura.

Miembros de la clase

- string m_IP
- string m_Hostname
- int m_iRoundTripTimeMs

Capítulo 14

TLuaPowershell

Ejecuta una cadena de comando Powershell. Devuelve un resultado de comando Powershell, un resultado de error, códigos de error y descripciones de error estándar.

Requisitos previos

La biblioteca TLuaPowershell utiliza WinRS (Shell remoto de Windows) para conectarse a un activo habilitado para Administración remota de Windows.

Información de Microsoft sobre WinRM/WinRS
(http://msdn.microsoft.com/en-us/library/aa384426%28v=vs.85%29.aspx):

"La Administración remota de Windows (WinRM) es la implementación de Microsoft del protocolo WS-Management, un protocolo estándar basado en el protocolo simple de acceso a objetos (SOAP) compatible con firewall que permite la interoperación de hardware y sistemas operativos de distintos proveedores.

Puede usar objetos de scripting de WinRM, la herramienta de línea de comandos de WinRM o la herramienta WinRS de línea de comandos de Shell remoto de Windows para obtener datos de administración de equipos locales y remotos que pueden tener controladores de administración de placa base (BMC). Si en la computadora se ejecuta una versión de sistema operativo basado en Windows que incluye WinRM, el Instrumental de administración de Windows (WMI) proporciona los datos de administración".

Para habilitar WinRM en un activo, escriba lo siguiente en el símbolo del sistema:

```
winrm /quickconfig
```

Puede ejecutar comandos Powershell o ejecutar scripts, pero recuerde que los comandos o los scripts que se ejecuten mediante WinRS no deben tener dependencias de interfaces de usuario. Por ejemplo, no puede ejecutar comandos que le soliciten "presionar cualquier tecla" en la consola local o que requieran cualquier otra respuesta interactiva.

En este capítulo

Script de ejemplo: TLuaPowershell

```
--executes a Powershell command string

bExitStatus = false
PS = TLuaPowershell()
bStatus = PS:Open(5985, false)

if bStatus == true then
  sCmd = "Get-Date\n"
  bExec = PS:ExecuteCommand(sCmd)
  if bExec == true then
        sStatus = PS:GetStdOut()
        bExitStatus = true
  else
        sStatus = "Execute failed. " .. PS:GetStdErr()
  end
else
  sStatus = "Failed to open connection. "  .. PS:GetErrorDescription()
end

SetExitStatus(sStatus, bExitStatus)
```

Script de ejemplo: TLuaPowershell (Windows Scripting)

```
--Create a script file named test.vbs in your C:\temp folder
--(on the remote asset) for this sample to work.
--The file should look like this:

strFile = WScript.Arguments(0)
Set objFSO = CreateObject("Scripting.FileSystemObject")
Set objFile = objFSO.GetFile(strFile)
strFileSize = objFile.Size
WScript.Echo strFile & " is " & strFileSize & " bytes."

--The script takes one parameter. That is the path to a file
--used for checking the file size.

bExitStatus = false
PS = TLuaPowershell()
bStatus = PS:Open(5985, false)

if bStatus == true then
  sCmd = "cscript /Nologo C:\\temp\\test.vbs \n"
  bExec = PS:ExecuteCommand(sCmd)
  if bExec == true then
        sStatus = PS:GetStdOut()
        bExitStatus = true
  else
        sStatus = "Execute failed. " .. PS:GetStdErr()
  end
else
  sStatus = "Failed to open connection. "  .. PS:GetErrorDescription()
end

SetExitStatus(sStatus, bExitStatus)
```

Abra

bool Open(unsigned short _iPort,bool bSecure=true,int dwWait=2500,const char *_pWorkingDir=nullptr)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando GetStdErr().

Parámetros

- iPort: puerto TCP que utiliza la administración remota de Windows (WinRM).
- bSecure: cuando se utiliza SSL, está establecido en verdadero (predeterminado); está establecido en falso para las conexiones que no son SSL.
- dwWait: el valor de tiempo de espera para una conexión correcta.

ExecuteCommand

ExecuteCommand(const char *pCommand)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetStdErr.

Parámetros

- pCommand: la cadena de comando Powershell.

GetStdOut

string GetStdOut(void)

Valores de devolución

Devuelve el resultado estándar del host remoto.

GetStdErr

string GetStdErr(void)

Valores de devolución

Devuelve el resultado de error estándar del host remoto.

GetErrorDescription

string GetErrorDescription(void)

Valores de devolución

Devuelve la descripción del error más reciente generada por el host remoto como una cadena.

GetErrorCode

dWord GetErrorCode(void)

Valores de devolución

Devuelve el código de error más reciente generado por el host remoto como una cadena.

Capítulo 15

TLuaRegistry

Esta clase proporciona acceso al Registro de Windows. Cuando se trabaja con el registro, en la documentación se utilizan dos términos importantes.

- Clave: una entidad en el subárbol de registro que puede contener claves y valores secundarios.
- Valor: una entidad sin entradas secundarias que contiene datos. Los datos pueden ser de diferentes tipos. Los tipos compatibles con esta implementación son datos en cadena, enteros y binarios.

Todas las operaciones del registro están relacionadas con el contexto en el que se ejecuta el script. Existe sólo una excepción, y se da cuando la clase se inicializa en el valor verdadero; en ese caso, todas las operaciones son relativas al equipo host de **Network Monitor**. Para obtener más información, consulte el script de ejemplo no. 2 en esta clase.

En este capítulo

Script de ejemplo: TLuaRegistry

```
--Demonstrates the Lua Windows Registry interface
--Open the registry on the host determined by the current context
Reg = TLuaRegistry();
if Reg:Open(Reg.LOCAL_MACHINE,"SOFTWARE\\Kaseya KNM") == true then sValue =
"";
bOK,sValue = Reg:ReadValue("INSTALL_SHORTCUTFOLDER",sValue);
SetExitStatus("KNM install path is -> "..sValue,bOK);
else
SetExitStatus("Could not open registry location",false);
end
```

BeginEnumValue

BeginEnumValue()

Comentarios

Se debe invocar esta función antes de la primera invocación de EnumValue(). Esta función asegura que EnumValue () comience en la parte superior de la lista de valores. Si no se invoca esta función antes de EnumValue(), se producen resultados impredecibles.

Cerrar

Close()

Comentarios

La función cierra la conexión al registro actualmente abierta.

Crear

bool Create(string sKey)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar una descripción del error invocando GetErrorDescription().

Parámetros

- sKey: una clave por crear.

Comentarios

La función Create() crea la clave del registro especificada. Si la clave ya existe en el registro, la función la abre. Esta función puede usarse para crear varias claves a la vez. Por ejemplo, se puede crear una subclave de tres niveles con un script al especificar una cadena de la siguiente forma:

subclave1\subclave2\subclave3

DeleteValue

bool DeleteValue(string sValueName)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar una descripción del error invocando GetErrorDescription().

Parámetros

- sValueName: el nombre de un valor por eliminar.

Comentarios

Esta función elimina un valor en la clave actual. Si el valor no existe, se produce un error en la función.

EnumValue

bool EnumValue(string &sValueName)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar una descripción del error invocando GetErrorDescription().

Parámetros

- sValueName: el nombre del siguiente valor enumerado en la clave actual.

Comentarios

Invoque esta función hasta que devuelva un resultado falso para enumerar todos los valores en la clave actual. Antes de invocar esta función por primera vez, se debe invocar BeginEnumValue().

Ejemplo 1

```
--KNM Lua API example (C) 2007 Kaseya AB
--Demonstrates the Lua Windows Registry interface
Reg = TLuaRegistry();
--Open the key to enumerate
if Reg:Open(Reg.LOCAL_MACHINE,"SOFTWARE\\Kaseya") == true then
  Reg:BeginEnumValue();
  bOk = true;
  repeat
        sValue = "";
        bOk,sValue = Reg:EnumValue(sValue);
        if bOk then print(sValue); end;
  until bOk == false;
else
  print("Failed to open the key");
end
```

GetErrorDescription

string GetErrorDescription()

Valores de devolución

Devuelve una cadena que describe el último error detectado cuando se invoca cualquier función de la clase.

Abra

bool Open(int iKey,string sKey)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar una descripción del error invocando GetErrorDescription().

Parámetros

- iKey: una clave que representa uno de los subárboles de registro.
- bCreate: una subclave en el subárbol de registro seleccionado.

Comentarios

La función abre una clave del registro en el subárbol de registro seleccionado. Tenga en cuenta que las credenciales del proceso (ya sea IDE o **Network Monitor**) pueden restringir el acceso a ciertas claves y subárboles. Las siguientes constantes están definidas para iKey:

- CLASSES_ROOT
- CURRENT_CONFIG
- CURRENT_USER
- LOCAL_MACHINE
- PERFORMANCE_DATA
- USERS

ReadValue

bool ReadValue(string sValueName,string &sData)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar una descripción del error invocando GetErrorDescription().

Parámetros

- sValueName: el nombre del valor por recuperar.
- sData: los datos que devuelve la función.

Comentarios

La función devuelve los datos del valor con el nombre especificado. Si el tipo de valor no es una cadena, se produce un error en la función.

ReadValue

bool ReadValue(string sValueName,int &iData)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar una descripción del error invocando GetErrorDescription().

Parámetros

- sValueName: el nombre del valor por recuperar.
- iData: los datos que devuelve la función.

Comentarios

La función devuelve los datos del valor con el nombre especificado. Si el tipo de valor no es un entero, se produce un error en la función.

ReadValue

string ReadValue(string sValueName,int &iSize)

Valores de devolución

Se devuelven los datos almacenados en el valor del registro. Si se produce un error en la función, se devuelve una cadena vacía. Se puede recuperar una descripción del error invocando GetErrorDescription().

Parámetros

- sValueName: el nombre del valor por recuperar.
- iSize: tamaño de los datos que devuelve la función, en bytes.

Comentarios

La función devuelve los datos del valor con el nombre especificado. Si el tipo de valor no es un entero, se produce un error en la función. El tamaño de los datos devueltos se almacena en el parámetro iSize. Si se produce un error en la función, el parámetro iSize se establece en cero.

SetValue

bool SetValue(string sValueName,string sData,int iDataSize)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar una descripción del error invocando GetErrorDescription().

Parámetros

- sValueName: el nombre del valor por escribir.
- sData: los datos que se escribirán en el valor.
- iSize: tamaño de los datos por escribir, en bytes.

Comentarios

La función escribe los datos especificados en el valor.

SetValue

bool SetValue(string sValueName,string sString)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar una descripción del error invocando GetErrorDescription().

Parámetros

- sValueName: el nombre del valor por escribir.
- sString: la cadena que se escribirá en el valor.
- iSize: tamaño de los datos por escribir, en bytes.

Comentarios

La función escribe la cadena especificada en el valor. Si el valor no existe, se produce un error en la función.

SetValue

bool SetValue(string sValueName,int iValue)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar una descripción del error invocando GetErrorDescription().

Parámetros

- sValueName: el nombre del valor por escribir.
- iValue: el entero que se escribirá en el valor.

Comentarios

La función escribe el entero especificado en el valor. Si el valor no existe, se produce un error en la función.

SetValueExpandedString

bool SetValueExpandedString(string sValueName,string sString)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar una descripción del error invocando GetErrorDescription().

Parámetros

- sValueName: el nombre del valor por escribir.
- sString: la cadena que se escribirá en el valor.

Comentarios

La función opera como la función SetValue normal, con una excepción. La cadena escrita puede contener referencias sin expandir a las variables de entorno (por ejemplo, “%PATH%”).

Capítulo 16

TLuaSFTPClient

Esta clase implementa una clase de cliente SFTP básico.

En este capítulo

Script de ejemplo: TLuaSFTPClient

```
--Demonstrates the Lua SFTP client class
--Create the client asset
sftp = TLuaSFTPClient()

--Connect to the remote SFTP server
if sftp:Connect("username","password") == false then
  SetExitStatus("No respons",false)
  return
end

--Create a directory handle and open the current directory
hHandle = TLuaSFTPClientDirectoryHandle()
bOk = sftp:OpenDir(".",hHandle)
if bOk == true then

  -- List the directory
  bOk = sftp:ListDir(hHandle)
  if bOk == false then
        SetExitStatus("Cannot list directory",false)
        sftp:CloseDir(hHandle)
        return
  end

  --Loop over the entries in the directory
  File = TLuaSFTPClientFile()
  while hHandle:Next(File) ~= false do
        --Print the file name
        print(File.m_sFilename)
  end
end
--Close handle
sftp:CloseDir(hHandle)
```

Cerrar

bool Close(TLuaSFTPClientHandle FileHandle)

Valor de devolución

Devuelve un resultado verdadero si la operación fue satisfactoria; de lo contrario, devuelve un resultado falso.

Parámetros

- FileHandle: identificador del archivo abierto anteriormente.

CloseDir

bool CloseDir(TLuaSFTPClientDirectoryHandle Handle);

Valor de devolución

Devuelve un resultado verdadero si la operación fue satisfactoria; de lo contrario, devuelve un resultado falso. En una operación satisfactoria, se cierra el identificador TLuaSFTPClientDirectoryHandle.

Parámetros

- Handle: identificador abierto por la función OpenDir.

Conectar

bool Connect(const unsigned int iPort=22,const unsigned short dwTimeout=25000);

Valor de devolución

Devuelve un resultado verdadero si la operación de conexión fue satisfactoria; de lo contrario, devuelve un resultado falso.

Parámetros

- sUsername: nombre de usuario.
- sPassword: contraseña.
- iPort: número de puerto donde el servidor escucha. El valor predeterminado es 22.
- iTimeout: el tiempo de espera en milisegundos que se debe esperar que el servidor responda. El valor predeterminado es 25000 (25 segundos).

CreateFile

bool CreateFile(string sPath,TLuaSFTPClientHandle hHandle)

Valor de devolución

Devuelve un resultado verdadero si se creó el archivo; si se produjo un error en la operación, devuelve un resultado falso. TLuaSFTPClientHandle contiene una referencia al archivo abierto si la operación es satisfactoria.

Parámetros

- sPath: ruta de acceso completa del archivo por crear. Deben existir los directorios incluidos en la ruta de acceso. De lo contrario, se producirá un error en la operación.
- hHandle: el identificador para crear el archivo.

Comentarios

El nuevo archivo creado tiene derechos de acceso de lectura y escritura.

ListDir

bool ListDir(TLuaSFTPClientDirectoryHandle Handle);

Valor de devolución

Devuelve un resultado verdadero si la operación fue satisfactoria; de lo contrario, devuelve un resultado falso. En una operación satisfactoria, los datos están listos para su recuperación en la clase TLuaSFTPClientDirectoryHandle.

Parámetros

- Handle: identificador abierto por la función OpenDir.

MkDir

bool MkDir(string sPath)

Valor de devolución

Devuelve un resultado verdadero si la operación fue satisfactoria; de lo contrario, devuelve un resultado falso.

Parámetros

- sPath: la ruta de acceso al directorio por crear; incluye el nombre del directorio nuevo.

Comentarios

Esta función no puede crear nuevos directorios de manera recursiva. Deben existir todos los directorios primarios del último directorio en la ruta de acceso.

OpenDir

bool OpenDir(string sPath,TLuaSFTPClientDirectoryHandle &Handle)

Valor de devolución

Devuelve un resultado verdadero si la operación fue satisfactoria; de lo contrario, devuelve un resultado falso.

Parámetros

- sPath: la ruta de acceso al directorio que se desea abrir.
- Handle: identificador que se devuelve para usar en operaciones subsiguientes.

Comentarios

Esta función “abre” un directorio para enumerar su contenido con la función ListDir.

Open_ForRead

bool Open_ForRead(string _sPath,TLuaSFTPClientHandle hHandle)

Valor de devolución

Devuelve un resultado verdadero si el archivo se abrió correctamente; si se produjo un error en la operación, devuelve un resultado falso.

Parámetros

- sPath: ruta de acceso completa del archivo.
- hHandle: identificador del archivo que se usa en operaciones subsiguientes.

Open_ForWrite

bool Open_ForWrite(string _sPath,TLuaSFTPClientHandle hHandle)

Valor de devolución

Devuelve un resultado verdadero si el archivo se abrió correctamente; si se produjo un error en la operación, devuelve un resultado falso.

Parámetros

- sPath: ruta de acceso completa del archivo.
- hHandle: identificador del archivo que se usa en operaciones subsiguientes.

Open_ForAppend

bool Open_ForAppend(string _sPath,TLuaSFTPClientHandle hHandle)

Valor de devolución

Devuelve un resultado verdadero si el archivo se abrió correctamente; si se produjo un error en la operación, devuelve un resultado falso.

Parámetros

- sPath: ruta de acceso completa del archivo. hHandle: identificador del archivo que se usa en operaciones subsiguientes.

Comentarios

Open_ForAppend abre el archivo en modo de escritura. La diferencia entre esta función y Open_ForWrite es que todos los datos se escriben al final del archivo, incluso si el puntero a archivo se reposiciona entre dos escrituras.

Leer

bool Read(TLuaSFTPClientHandle FileHandle,int iOffset,int iLen,string &sData)

Valor de devolución

Devuelve un resultado verdadero si la operación fue satisfactoria; de lo contrario, devuelve un resultado falso.

Parámetros

- FileHandle: identificador del archivo abierto anteriormente.
- iOffset: desplazamiento en bytes desde donde leer el archivo.
- iLen: longitud de datos por leer.
- sData: variable donde se colocan datos.

Comentarios

Sólo se pueden leer archivos de texto con esta función.

Ejemplo

```
--KNM Lua API example (C) 2010 Kaseya AB
--Demonstrates the Lua SFTP client class

sftp = TLuaSFTPClient()
hFileHandle = TLuaSFTPClientHandle()

--Open the file
bOk = sftp:Open_ForRead("test.txt",hFileHandle)
if bOk == false then
  SetExitStatus("Open failed",false)
  return
end

sTemp = ""
--Read the first 20 bytes
bOk,sTemp = sftp:Read(hFileHandle,0,20,sTemp)
if bOk == false then
  SetExitStatus("Read failed",false)
  return
end
print(sTemp)
```

Remover

bool Remove(string sPath);

Valor de devolución

Devuelve un resultado verdadero si la operación fue satisfactoria; de lo contrario, devuelve un resultado falso.

Parámetros

- sPath: la ruta de acceso del archivo que se desea quitar.

Renombrar

bool Rename(string sPath,string sNewPath);

Valor de devolución

Devuelve un resultado verdadero si la operación fue satisfactoria; de lo contrario, devuelve un resultado falso.

Parámetros

- sPath: la ruta de acceso del archivo existente al que se le desea cambiar el nombre.
- sNew: la ruta de acceso con el nuevo nombre de archivo.

RmDir

bool RmDir(string sPath)

Valor de devolución

Devuelve un resultado verdadero si la operación fue satisfactoria; de lo contrario, devuelve un resultado falso.

Parámetros

- sPath: la ruta de acceso al directorio que se desea eliminar.

Comentarios

Esta función sólo elimina directorios vacíos.

Escribir

bool Write(TLuaSFTPClientHandle FileHandle,const int iOffset,string vData)

Valor de devolución

Devuelve un resultado verdadero si la operación fue satisfactoria; de lo contrario, devuelve un resultado falso.

Parámetros

- FileHandle: identificador del archivo abierto anteriormente.
- iOffset: desplazamiento en bytes desde donde escribir en el archivo.
- sData: la cadena de texto por escribir.

Ejemplo

```
--KNM Lua API example (C) 2010 Kaseya AB
--Demonstrates the Lua SFTP client class

sftp = TLuaSFTPClient()
hFileHandle = TLuaSFTPClientHandle()

--Open the file
hFileHandle = TLuaSFTPClientHandle()
if sftp:Open_ForWrite("test.txt",hFileHandle) == false then
  SetExitStatus("Open of file failed",false)
  return
end

--Create a string and write it to the begining of the file
sString = [[ test text ]];
if sftp:Write(hFileHandle,0,sString) == false then
  SetExitStatus("Write failed",false)
  return
end

--Close the file
sftp:Close(hFileHandle)
```

Capítulo 17

TLuaSFTPClientAttributes

Esta clase contiene atributos que describen un directorio o un archivo recuperado por la función ListDir.

En este capítulo

AccessedTime

bool AccessedTime(TLuaDateTime &Time)

Valor de devolución

Devuelve un resultado verdadero si el valor está presente; de lo contrario, devuelve un resultado falso.

Parámetros

- Time: contiene la hora en que se accedió al archivo por última vez.

CreatedTime

bool CreatedTime(TLuaDateTime &Time)

Valor de devolución

Devuelve un resultado verdadero si el valor está presente; de lo contrario, devuelve un resultado falso.

Parámetros

- Time: contiene la hora en que se creó el archivo.

máq.

bool Group(string &sGroup)

Valor de devolución

Devuelve un resultado verdadero si el valor está presente; de lo contrario, devuelve un resultado falso.

Parámetros

- sOwner: contiene el nombre del grupo del archivo o directorio.

ModifiedTime

bool ModifiedTime(TLuaDateTime &Time)

Valor de devolución

Devuelve un resultado verdadero si el valor está presente; de lo contrario, devuelve un resultado falso.

Parámetros

- Time: contiene la hora en que se modificó el archivo por última vez.

Propietario

bool Owner(string &sOwner)

Valor de devolución

Devuelve un resultado verdadero si el valor está presente; de lo contrario, devuelve un resultado falso.

Parámetros

- sOwner: contiene el nombre del propietario del archivo o directorio.

PermissionBits

bool PermissionBits(int &iPermissionsBits)

Valor de devolución

Devuelve un resultado verdadero si el valor está presente; de lo contrario, devuelve un resultado falso.

Parámetros

- iPermissionsBits: contiene un valor decimal que representa el permiso del archivo o el directorio.

Tamaño

bool Size(int &iBytesHighDWord,int &iBytesLowDWord);

Valor de devolución

Devuelve un resultado verdadero si el valor está presente; de lo contrario, devuelve un resultado falso.

Parámetros

- iBytesHighDWord: contiene la porción alta de dword del entero de 64 bits.
- iBytesLow DWord: contiene la porción baja de dword del entero de 64 bits.

Comentarios

El tamaño del archivo se informa en bytes como un entero de 64 bits. Dado que Lua no cuenta con un tipo de datos entero de 64 bits, la información se separó en dos enteros de 32 bits. Si el tamaño del archivo es de menos de 2 GB, iBytesHighDWord siempre es cero.

SizeMB

bool SizeMB(unsigned int &iSizeMB);

Valor de devolución

Devuelve un resultado verdadero si el valor está presente; de lo contrario, devuelve un resultado falso.

Parámetros

- iSizeMB: contiene el tamaño del archivo en megabytes.

Comentarios

Se proporciona como una alternativa fácil de usar a la función Size(). Devuelve el tamaño del archivo redondeado hacia abajo.

Capítulo 18

TLuaSFTPClientDirectoryHandle

Esta clase se usa junto con las funciones OpenDir, ListDir y CloseDir.

En este capítulo

Siguiente

bool Next(TLuaSFTPClientFile &hFile)

Valor de devolución

Devuelve un resultado verdadero si la función TLuaSFTPClientFile proporcionada contiene datos.

Comentarios

Repita la función hasta que devuelva un resultado falso para recuperar toda la información devuelta por la función ListDir.

Capítulo 19

TLuaSFTPClientFile

TLuaSFTPClientFile es una clase de sólo lectura.

Miembros de la clase

- string m_sFilename
- string m_sLongFilename
- TLuaSFTPClientAttributes m_Attribs

Capítulo 20

TLuaSNMP

Esta clase implementa un cliente SNMP básico que puede realizar operaciones set y get.

En este capítulo

Script de ejemplo: TLuaSNMP

```
--Simple example of SNMP interface
SNMP = TLuaSNMP();
SNMP:Open("public");

iSyntax =1

sData = SNMP:Get("iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.
ifEntry.ifInOctets.1",iSyntax);

if sData ~= "" then

  print(sData);
  SetExitStatus("Got sample value: "..sData.." bytes received",true);

else

  SetExitStatus("Get failed",false);

end
```

BeginWalk

BeginWalk(string sOID)

Parámetros

- sOID: OID que representa el comienzo de un examen del árbol de OID. Ejemplo de OID: iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable

Comentarios

Antes de la primera invocación de la función Walk, el programa debe invocar la función BeginWalk para establecer el comienzo de Walk. Walk recupera todos los identificadores de activos secundarios y del mismo nivel del OID inicial establecidos por las funciones BeginWalk.

Cerrar

Close()

Comentarios

Cierra la conexión SNMP.

Get

string Get(string sOID,int iSyntax)

Valores de devolución

Una cadena con el valor obtenido del agente SNMP remoto.

Parámetros

- sOID: OID para usar en la operación Get. Cuando se consulta una interfaz, se puede utilizar el usuario @ para especificar el índice de interfaz.

  Ejemplo del uso del usuario @:

```
  iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifInOctets@NV
IDIA nForce Networking Controller
```

  Ejemplo del OID normal

```
  iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifInOctets.1
```

- iSyntax: especifica el formato de los datos devueltos. Puede ser una de las siguientes constantes:
- SNMP_NOSYNTAX
- SNMP_IPADDRESS
- SNMP_INTEGER
- SNMP_UNSIGNED32
- SNMP_COUNTER32
- SNMP_GAUGE32
- SNMP_TIMETICKS
- SNMP_OPAQUE
- SNMP_OCTETSTRING
- SNMP_DATA_AS_HEXSTRING

Lectura de valores binarios

Algunos OID pueden devolver datos binarios en lugar de, por ejemplo, una cadena o un entero. Esto puede suponer un problema, dado que la función Get devuelve una cadena terminada en cero. Una solución para este problema es establecer la variable iSyntax en SNMP_DATA_AS_HEXSTRING. De esta manera, la función devuelve los datos binarios cifrados en formato hexadecimal.

Ejemplo de tres bytes cifrados en formato hexadecimal

```
49 4E4D
```

Abra

bool Open(string sCommunity, int iPort=161)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- sCommunity: el nombre de la comunidad, por lo general, público.
- iPort (optativo): especifique el número de puerto si necesita usar un puerto distinto del estándar (puerto 161).

Establecer

bool Set(string sOID,string sData,int iSyntax)

Valores de devolución

Valor distinto de cero si la función se ejecutó correctamente; de lo contrario, devuelve cero.

Parámetros

- sOID: OID para usar en la operación set.
- sData: los datos textuales que se usan en la operación set.
- iSyntax: especifica el formato del parámetro sData. Puede ser una de las siguientes constantes:
  - ✓ SNMP_NOSYNTAX
  - ✓ SNMP_IPADDRESS
  - ✓ SNMP_INTEGER
  - ✓ SNMP_UNSIGNED32
  - ✓ SNMP_COUNTER32
  - ✓ SNMP_GAUGE32
  - ✓ SNMP_TIMETICKS
  - ✓ SNMP_OPAQUE
  - ✓ SNMP_OCTETSTRING

Walk

TLuaSNMPResult TLuaSNMP::Walk()

Valores de devolución

Devuelve el identificador del próximo OID en la variable miembro m_sOID en la estructura TLuaSNMPResult. No devuelve la cadena del OID completa. Cuando se alcanza el final, el miembro m_sOID de la estructura TLuaSNMPResult está vacío.

Comentarios

Antes de la primera invocación de la función Walk, el programa debe invocar la función BeginWalk para establecer el comienzo de Walk. Walk recupera todos los identificadores secundarios y de objetos del OID inicial establecidos por las funciones BeginWalk.

Ejemplo

```
--KNM Lua API example (C) 2007 Kaseya AB
--Simple example of SNMP interface
SNMP = TLuaSNMP();
SNMP:Open("public");
sOID = "iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry";

--A repeat ... until loop
Result = TLuaSNMPResult();
SNMP:BeginWalk(sOID);
repeat
  Result = SNMP:Walk(sOID);
  sOID = Result.m_sOID;
  print("OID "..sOID);
  print("Data "..Result.m_sData);
  print("Syntax "..Result.m_iSyntax);
until Result.m_sOID == "";
```

TLuaSNMPResult

TLuaSNMPResult es una clase de sólo lectura devuelta por la función Walk. Si se modifica, se inicia

una excepción.

Miembros de la clase

- string m_sOID
- string m_sData
- int m_iSyntax

Capítulo 21

TLuaSSH2Client

Esta clase implementa un cliente SSH 2.0 que puede ejecutar comandos en un servidor remoto.

En este capítulo

Script de ejemplo: TLuaSSH2Client

```
SSHClient = TLuaSSH2Client();
SSHClient:Open(23,"testuser","testpassword");
if SSHClient:ExecuteCommand("shutdown") == true then
  print(SSHClient:GetStdOut());
  SetExitStatus("Exec ok",true);
else
  print(SSHClient:GetStdErr());
  print(SSHClient:GetErrorDescription());
  SetExitStatus("Exec failed",true);
end
```

ExecuteCommand

bool ExecuteCommand(string sCommand,DWORD dwWait/*=2500*/)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- sCommand: la cadena con el comando por ejecutar en el host remoto.
- dwWait (optativo): el tiempo que hay que esperar para que finalice la ejecución. El valor predeterminado es 25 segundos.

GetErrorDescription

string GetErrorDescription(void)

Valores de devolución

Devuelve la descripción del error más reciente generada por el cliente como una cadena.

GetStdErr

string GetStdErr(void)

Valores de devolución

Devuelve el resultado de error estándar del host remoto.

GetStdOut

string GetStdOut(void)

Valores de devolución

Devuelve el resultado estándar del host remoto.

Abra

bool Open(int iPort)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función GetErrorDescription. Si se produce un error como resultado del comando, se puede recuperar más información invocando GetStdErr.

Parámetros

- iPort: puerto SSH. El puerto predeterminado es 23.
- sUsername: nombre de usuario.
- sPassword: contraseña.

Capítulo 22

TLuaSocket

Esta clase proporciona operaciones de socket básicas. Los sockets se pueden abrir en modo UDP o TCP.

En este capítulo

Script de ejemplo: TLuaSocket

```
--Construct a new socket device
socket = TLuaSocket()
iPortNumber = 8080

--Open a TCP socket
iRet = socket:OpenTCP(iPortNumber)

--If OpenTCP returned a 0 (boolean FALSE) then the open failed
if iRet==0 then

  print("Cannot open port "..iPortNumber.." Error code:"..GetLastError())

else

  --Read some data (max 1024 bytes) from the socket
  iReadSize = 1024
  data = socket:Read(iReadSize)

  --Print the content
  if iReadSize > 0 then
        print("Data received from server:\n\n")
        print(data)
  else
        print("No data received from server")
  end

end
socket:Close()
```

Cerrar

int Close()

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Comentarios

Cierra el socket abierto anteriormente con OpenTCP u OpenUDP.

OpenTCP

int OpenTCP(int iPort)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- iPort: el puerto por abrir.

Comentarios

Abre un socket TCP con el número de puerto especificado.

OpenUDP

int OpenUDP(int iPort)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- iPort: un puerto específico para usar con el socket.

Comentarios

Abre un socket UDP con el número de puerto especificado.

Leer

string Read(int iSize,int iTimeout=1)

Valores de devolución

Si la función se ejecutó correctamente, devuelve una matriz de datos. De lo contrario, si no se puede leer ningún dato, no devuelve nada. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- iSize: cuando la invocación de la función devuelve la variable, se establece en el tamaño de los datos leídos. Si no se leyeron datos, este valor es cero.
- iTimeout: la cantidad de tiempo en segundos que se espera que los datos lleguen al socket. El valor predeterminado es un segundo.

Comentarios

La función sólo bloquea la ejecución durante la cantidad de tiempo especificada por el valor de tiempo de espera. Si no se reciben datos durante ese período, la función devuelve un valor nulo.

Escribir

int Write(string Data,int iSize)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- sData: una matriz con datos por enviar.

- iSize: el tamaño de los datos en la matriz.

Capítulo 23

TLuaSocketSecure

Esta clase proporciona operaciones básicas de socket seguro, lo que se suele denominar “Seguridad de la capa de transporte” (TLS) o “Capa de sockets seguros” (SSL) (su predecesora).

En este capítulo

Script de ejemplo: TLuaSocketSecure

```
--This function is called by KNM when enumerating a field
function OnEnumerate(sFieldToEnum)

  Enum = LuaScriptEnumResult()

  if sFieldToEnum == "Ignore connection problems" then
        Enum:Add("Yes")
        Enum:Add("No")
  end

  return Enum
end

--This function is called by KNM to retrieve a script configuration
function OnConfigure()

  Config = LuaScriptConfigurator()
  Config:SetAuthor("Robert Aronsson, Kaseya AB")
  Config:SetDescription("The script check if a certificate is about to
expire within the configured number of days.")
  Config:SetMinBuildVersion(5280)
  Config:SetScriptVersion(1,0)

  Config:AddArgument("Port number","Port number to connect on",
LuaScriptConfigurator.CHECK_NOT_EMPTY)
  Config:AddArgument("Number of days","Check if certificate expres within
this period",LuaScriptConfigurator.CHECK_NOT_EMPTY)
  Config:AddArgument("Ignore connection problems","Do you want thescript to
report connection problems as well ?",LuaScriptConfigurator.ENUM_AVAIL +
LuaScriptConfigurator.CHECK_NOT_EMPTY)

  Config:SetEntryPoint("main")

  return Config
end

--This is the entry point
function main()

  local iPort = GetArgument(0)
  local iNumDays = GetArgument(1)
  local bReportConnectionProblem = false;
  if GetArgument(2) == "Yes" then
        bReportConnectionProblem = true
  end

  --Timeperiod that the certificate should be valid within
  local iOffsetTime = (60 * 60 * 24) * iNumDays

  --Default values for test eval
  local bTestOk = true;
  local sText = "Certificate ok";
```

```
  --Open socket
  Socket = TLuaSocketSecure()
  if Socket:Open(iPort) ~= 0 then

        CurrentTime = TLuaDateTime();

        --The time was retrieved during the connect
        Time = Socket:GetCertificateExpiryDate();

        print("Certificate expires ("..Time:GetDate() .." " .. Time:
GetTime()..")");

        --Check time
        iExpiryTime = Time:Get() -iOffsetTime;
        if Time:Get() < CurrentTime:Get() then
              bTestOk = false;
              sText = "Certificate have already expired ("..Time: GetDate()
.." " .. Time:GetTime()..")";
        else
              if iExpiryTime < CurrentTime:Get() then
                    bTestOk = false;
                    sText = "Certificate is about to expire in less than
"..iNumDays.." days"
              end
        end
  else
        --Failed to open the socket, server down ?
        if bReportConnectionProblem == true then
              bTestOk = false;
        end
        sText = "Cannot connect to host.";
  end

  --Report status and exit
  SetExitStatus(sText,bTestOk);
end
```

Abra

int Open(int iPort)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- iPort: el puerto por abrir.

Cerrar

int Close()

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Leer

string Read(int iSize)

Valores de devolución

Si la función se ejecutó correctamente, devuelve una matriz de datos. De lo contrario, si no se puede leer ningún dato, no devuelve nada. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- iSize: cuando la invocación de la función devuelve la variable, se establece en el tamaño de los datos leídos. Si no se leyeron datos, este valor es cero.

Escribir

int Write(string Data,int iSize)

Valores de devolución

Si la función se ejecutó correctamente, devuelve un valor distinto de cero; de lo contrario, devuelve cero. Se puede recuperar un código de error específico invocando la función global GetLastError.

Parámetros

- sData: una matriz con datos por enviar.
- iSize: el tamaño de los datos en la matriz.

GetCertificateExpiryDate

TLuaDateTime GetCertificateExpiryDate()

Valores de devolución

Una estructura TLuaDateTime que contiene la fecha de caducidad del certificado del host remoto. Si se produce un error en la invocación de Connect(), la estructura contiene una fecha en cero.

Comentarios

Esta función se puede usar para determinar si un certificado está por caducar o ya caducó.

Capítulo 24

TLuaStorage

La clase proporciona funciones para guardar datos textuales entre sesiones de script. Puede ser útil cuando se desea basar la iteración del script actual en un resultado previo o establecer una comunicación entre dos scripts que no están relacionados.

En este capítulo

CreateItem

bool CreateItem(string sName,string sKey,string sData=NULL,int iSize=0)

Valores de devolución

Valor distinto de cero si la función se ejecutó correctamente; de lo contrario, devuelve cero.

Parámetros

- sName: el nombre único del elemento. Si el nombre ya está creado, se produce un error en la función.
- sKey: el nombre de clave del elemento, que debe ser único. Si ya existe, se produce un error en la función.
- sData: los datos optativos que se relacionan con el elemento.
- iSize: el tamaño de los datos. Sólo se necesita si los datos se proporcionan con la función.

Comentarios

La función crea un elemento y un subelemento denominado “clave”. El usuario puede relacionar datos con esta clave. Los datos se pueden adquirir más adelante invocando la función FindItem.

UpdateItem

bool UpdateItem(string sName,string Key,string Data=NULL,int iSize=0)

Valores de devolución

Valor distinto de cero si la función se ejecutó correctamente; de lo contrario, devuelve cero.

Parámetros

- sName: el nombre único del elemento. Ya debe existir un elemento con este nombre.
- sName: el nombre de clave del elemento. Ya debe existir una clave con nombre.
- sData: los datos optativos que se relacionan con el elemento. Los datos reemplazan los datos actuales almacenados en el elemento (si los hubiera).
- iSize: el tamaño de los datos. Sólo se necesita si los datos se proporcionan con la función.

Comentarios

La función actualiza un elemento que ya está creado. Si la combinación de elemento y clave no existe, se produce un error en la función.

DeleteItem

void DeleteItem(string sName,string sKey);

Parámetros

- sName: el nombre del elemento.
- sKey: el nombre de la clave por eliminar.

Comentarios

La función elimina una combinación de elemento y clave. Los datos relacionados con la clave también se eliminan.

FindItem

TLuaStorageItem FindItem(string sName,string sKey);

Valores de devolución

Valor distinto de cero si la función se ejecutó correctamente; de lo contrario, devuelve cero.

Parámetros

- sName: el nombre único del elemento. Ya debe existir un elemento con este nombre.
- sName: el nombre de clave del elemento. Ya debe existir una clave con nombre.

Comentarios

La función recupera un elemento almacenado. La clase devuelta contiene los nombres de elemento y clave, así como los datos relacionados con el elemento.

TLuaStorageItem

TLuaStorageItem es una clase de sólo lectura. Si se modifica, se inicia una excepción.

Miembros de la clase

- string m_Key
- string m_Name
- string m_pData
- int m_iSize

Capítulo 25

TLuaTimer

Esta clase proporciona un temporizador con precisión de milisegundos.

En este capítulo

Script de ejemplo: TLuaTimer

```
--Demonstrates the Lua timer interface
Timer = TLuaTImer();
Timer:Start()
print("Timer started");
Wait(1000);
print("Operation took "..Timer:Stop().." ms";
```

Inicio

Start()

Comentarios

Esta función hace que comience a correr el tiempo, y una invocación subsiguiente de la función Stop() devuelve la hora entre las invocaciones de Start y Stop. Después de que se invoca Stop, invoque Start nuevamente para restablecer el temporizador e iniciar un nuevo período.

Detener

int Stop()

Valores de devolución

Se devuelve la cantidad de milisegundos desde la invocación de la función Start().

Capítulo 26

TLuaWinperf

Esta clase proporciona funciones para consultar valores numéricos en el registro de rendimiento de Windows. Se proporciona como una alternativa fácil de usar a la clase TLuaWMIQuery más avanzada. La clase se ejecuta en el contexto de seguridad del proceso o subproceso que inicia el script. En el IDE, el contexto de seguridad se hereda del escritorio. Cuando la ejecuta el monitor de script Lua, el contexto de seguridad se puede establecer seleccionando una cuenta predeterminada en la página de propiedades del monitor.

En este capítulo

Script de ejemplo: TLuaWinperf

```
--Prints the number of private bytes the notepad.exe application have
allocated
Perf = TLuaWinperf()
if Perf:Query("Process","Private Bytes","notepad") then
  Value = Perf:GetResult();
  print(Value);
else
  print(Perf:GetErrorDescription())
end
```

GetErrorDescription

string GetErrorDescription()

Valores de devolución

Devuelve una cadena que describe el último error detectado cuando se invoca cualquier función de la clase.

GetResult

double GetResult()

Valores de devolución

Devuelve un valor de contador numérico. Si se produce un error en la invocación previa de Query(), esta función devuelve cero.

Query

bool Query(string sDeviceName,string sCounterName,string sInstanceName=NULL);

Valores de devolución

El resultado es verdadero si la consulta se ejecutó correctamente; si ocurrió un error, el resultado es falso.

Parámetros

- sDeviceName: una cadena con el nombre del activo que contiene el contador por consultar.
- sCounterName: una cadena con el nombre del contador por consultar.
- sInstanceName (optativo): una cadena con el nombre de la instancia de contador.

Comentarios

Los nombres de activo, de contador y de instancia se pueden obtener en el monitor Winperf de **Network Monitor**; para ello, se debe hacer clic en el botón de enumeración o utilizar la aplicación perfmon.exe de Windows. Para recuperar el valor, invoque GetResult() después de que se completa esta función.

Capítulo 27

TLuaWMIQuery

Esta clase proporciona funciones para consultar las propiedades de WMI. La clase se ejecuta en el contexto de seguridad del proceso o subproceso que inicia el script. En el IDE, el contexto de seguridad se hereda del escritorio. Cuando la ejecuta el monitor de script Lua, el contexto de seguridad se puede establecer seleccionando una cuenta predeterminada en la página de propiedades del monitor. La cuenta debe tener habilitada la delegación.

En este capítulo

TLuaWMIQuery

```
--Demonstrates the Lua WMI interface
Query = TLuaWMIQuery(); Query:Execute("select Deviceid,Size,Freespace from
win32_logicaldisk"); print(Query:GetErrorDescription());

while (Query:NextInstance()) do
  sDeviceID = "";
  bOk,sDeviceID = Query:GetProperty("Deviceid",sDeviceID);
  print(sDeviceID);
end
```

Ejecutar

bool Execute(const char *pWQL)

bool Execute(const char *pWQL,const int iPrivacy)

Valores de devolución

El resultado es verdadero si la consulta se ejecutó correctamente; si ocurrió un error, el resultado es falso.

Parámetros

- sWQL: una cadena que contiene una consulta WQL.

Comentarios

Ejecuta una consulta WQL (lenguaje de consulta de WMI). Invoca Next(); se puede usar GetProperty() para recuperar el resultado.

GetErrorDescription

string GetErrorDescription()

Valores de devolución

Devuelve una cadena que describe el último error detectado cuando se invoca cualquier función de la clase. Es útil para la depuración de consultas de WMI.

GetProperty

bool,string GetProperty(string sPropertyName,string sReturnValue);

Valores de devolución

Si la función se ejecuta correctamente, devuelve un resultado verdadero y un valor en una cadena. Si se produce un error en la función, devuelve un resultado falso y una cadena vacía. Se puede recuperar información más detallada sobre el error invocando GetError().

Parámetros

- sPropertyName: el nombre de la propiedad por recuperar.

- sReturnValue: la cadena definida que recibe el valor de devolución. El valor de devolución siempre es una cadena, aunque el tipo de propiedad sea, por ejemplo, un entero o un número real.

Comentarios

Recupera un valor de propiedad en el resultado actual. Para recuperar el próximo valor de la misma propiedad, invoque la función NextInstance(). Si NextInstance() devuelve un resultado falso, no hay más valores.

NextInstance

bool NextInstance()

Valores de devolución

El resultado es verdadero si se obtiene un nuevo resultado. Si no existen más resultados para la consulta, el resultado es falso.

Comentarios

La función recupera un nuevo resultado generado por una invocación previa de la función Execute. Se debe invocar esta función antes de la primera invocación de la función GetProperty.

SetNamespace

SetNamespace(string sNamespace)

Parámetros

- sNamespace: cadena con espacio de nombres WMI para usar en todas las invocaciones futuras.

Comentarios

El espacio de nombres predeterminado que usa la clase TLuaWMIQuery es root\cvmi2.

Capítulo 28

TLuaXMLNode

La clase representa un elemento XML y puede contener uno o más elementos secundarios.

En este capítulo

FindAttribute

string FindAttribute(string sName)

Valores de devolución

La función devuelve una cadena con el valor del atributo. Si no se puede encontrar el atributo, la cadena que se devuelve está vacía.

Parámetros

- sName: el nombre del atributo.

FindChildNode

TLuaXMLNode FindChildNode(string sElementName, int iOffset)

Valores de devolución

Si se encuentra el elemento, la función devuelve un activo TLuaXMLNode válido.

Parámetros

- sElementName: el nombre del elemento secundario para este nodo.
- iOffset: un índice de base cero para recuperar elementos secundarios con el mismo nombre en el nodo.

Comentarios

Esta función se puede usar para iterar en una cantidad de elementos secundarios con el mismo nombre. Aumente el parámetro de desplazamiento para recuperar el siguiente elemento.

GetData

string GetData()

Valores de devolución

La función devuelve los datos en el elemento.

GetTag

string GetTag()

Valores de devolución

La función devuelve el nombre de etiqueta del elemento.

GetParentNode

TLuaXMLNode GetParentNode()

Valores de devolución

La función devuelve el elemento primario del elemento de documento XML actual.

IsValid

bool IsValid()

Valores de devolución

La función devuelve un resultado verdadero si el nodo es válido y un resultado falso si el nodo no es válido.

Comentarios

Todas las funciones de búsqueda devuelven un activo TLuaXMLNode. La función IsValid() se usa para determinar si la búsqueda fue satisfactoria.

Capítulo 29

TLuaXMLReader

Esta clase proporciona funcionalidad básica para analizar y recorrer documentos XML.

En este capítulo

FindChildNode

TLuaXMLNode FindChildNode(string sElementName, TLuaXMLNode ParentNode)

Valores de devolución

Si se encuentra el elemento, la función devuelve un activo TLuaXMLNode válido.

Parámetros

- sElementName: el nombre del elemento secundario de ParentNode.
- ParentNode: el nodo primario donde se desea buscar.

Comentarios

Tenga en cuenta que la función devuelve el primer elemento con el nombre especificado.

FindNode

TLuaXMLNode FindNode(string sElementName, TLuaXMLNode RootNode)

Valores de devolución

Si se encuentra el elemento, la función devuelve un activo TLuaXMLNode válido.

Parámetros

- sElementName: el nombre del elemento secundario de ParentNode.
- RootNode: el nodo primario para usar como punto de partida de la búsqueda.

Comentarios

La función busca el documento XML de forma recursiva con “RootNode” como punto de partida de la búsqueda.

FromXML

bool FromXML(string XML)

Valores de devolución

El resultado es verdadero si la operación se ejecutó correctamente; si ocurrió un error, el resultado es falso.

Parámetros

- sXML: un documento XML para analizar.

Comentarios

Tenga en cuenta que el analizador no valida el esquema del documento.

GetRootNode

TLuaXMLNode GetRootNode()

Valores de devolución

La función devuelve el elemento raíz del documento XML.

Índice

U

W