WS Functional Automation

8.13 wsCurl

Purpose

With wscurl(), you can transfer data from WS scripts to external URLs.

The wsCurl library is used with the load command to transfer data from a WS script to an external URL, such as Google Translate. This is particularly useful for foreign-language users who wish to input text in a non-Western language and then use a service such as Google Translate to change the text into English or some other language.

Desktop Implementations

To add the wsCurl functionality to an existing WS Desktop implementation, please do the following:

1. Obtain the wscurl.dll library from Synactive.
2. Place the wscurl.dll library in the following directory: C:\Program Files\SAP\FrontEnd\SAPgui.
3. Any scripts will be in the WS script folder. The default path is: C:\guixt.

Liquid UI Server Implementations

To add the wsCurl functionality to an existing Liquid UI Server implementation, please do the following:

1. Obtain the wscurl.dll library from Synactive.
2. Place the wscurl.dll library in the same directory as the sapproxy.exe file. This directory is usually found in the following location: C:\Program Files\Synactive Inc\GuiXTServer.
3. Any scripts will be in the WS script folder. The default path is: C:\guixt.

Offline Implementations

To add the wsCurl functionality to an existing Offline implementation, please do the following:

1. Obtain the wscurl.dll library from Synactive.
2. Place the wscurl.dll library in the Offline directory.
3. Any scripts will be in the Synscript folder.

Using wsCurl

The wsCurl feature is built using the Curl project. The syntax for this is as follows:

```
load('wscurl');
var ch = new wscurl(query_string);
ch->setopt(CURLOPT_HEADER, false);
var response = ch->exec()
```

```
ch->close();
ch = nil;
```

Please see the examples in Accessing Google Translate with wsCurl and Sending Webmail with wsCurl for examples on using wsCurl in WS scripts.

Examples

Sending Webmail with wsCurl

In the following example, we will demonstrate how to use the wsCurl functionality in WS to send Web mail.

1. Open the script file you will use to build the wsCurl function. You can create a new script file to call from the original script file if necessary.
2. Load the wsCurl files as shown in the following example:

   ```
   load('wscurl.dll');
   ```

3. Create a function. In our example, we will name the function 'translate'. All the code will be contained in this function. An example is shown below:

   ```
   function sendmail()
   { }
   ```

4. Set the applicable variables and identifiers as shown in the following list.
   Initialize the wscurl object

   ```
   var wsCurl = new Curl();
   ```

   Set Mailserver URL

   In this step, we will set the URL for the mail server. Note that in our example, we are using port 587 as opposed to the normal SMTP port 25. We are using 587 because we are using secure mail submission as defined in RFC4403. However, you should set your port to match the port on your mail server. The code is as follows:

   ```
   wsCurl.setopt(Curl.CURLOPT_URL, "
   smtp://smtp.gmail.com:587");
   ```

   Upgrade Connection to TLS

   In this step, we will upgrade the connection from plain text to

Transport Layer Security (TLS). To do this, we will use the STARTTLS command.

Note: We recommend using Curl.USESSL_ALL as shown below, not Curl.USESSL_TRY. This is because if the TLS upgrade fails, the transfer will proceed.

```
wsCurl.setopt(Curl.CURLOPT_USE_SSL,Curl.USESSL_ALL);
```

Disable Peer and Host Verification

There may be times when your server does not have a valid certificate. IN these cases, you can disable part of the TLS protection by using the following code to set the CURLOPT_SSL_VERIFYPEER and CURLOPT_SSL_VERIFYHOST options to 0.

```
wsCurl.setopt(Curl.CURLOPT_SSL_VERIFYPEER, 0);
wsCurl.setopt(Curl.CURLOPT_SSL_VERIFYPEER,0);
```

Note: We do not recommend disabling any part of your security, but having partial security is better than sending details in plain text. However, we recommend adding the issuer certificate (or the host certificate if the server certificate is self-signed), to the libcurl using the CURLOPT_CAINFO option.

To add the certificate to libcurl, use the following code:

```
wsCurl.setopt(Curl.CURLOPT_CAINFO,"
/path/to/certificate.pem");
```

Set Password and Username

The most common reason to have transport security is to protect your users'login credentials from theft. Use the following code to set user names and passwords.

```
wsCurl.setopt(CURLOPT_USERNAME,"user@example.com");
 wsCurl.setopt(CURLOPT_PASSWORD,"Password");
```

Set Envelope Reverse Path

Use the following code to set the envelope reverse path for the email. Please note that you must use brackets around the email address as shown in the example below. .

```
var translateLangIdentifier = "&target=";
```

Set

Set the target language code itself. This will be a two-digit code and in our example, we will use English. A complete list of available language codes can be found in the Language Code Reference.

```
var translateLang = "en";
```

Build the URL

The following code will create the complete URL that we will use to make the call for a translation request. This is demonstrated as follows:

```
var completeURL = baseURL + textToConvert + sourceLangIden
tifier + sourceLang + translateLangIdentifier + translateL
ang;
```

Make a Translation Request

Now that you have the complete URL, you can make a request for translation as shown in the below example:

Note: This example is for requests that return a JSON object in a string.

```
wsCurl.setopt(Curl.CURLOPT_URL,completeURL);
```

Call the Execute function

The final step in the process is to call the execute function. This will dispatch a response to your request. An example is shown below:

Note: You can check the return value for errors with the error codes that are explained in the Error Codes section. A return value of 0 means the operation succeeded.

```
var response = wsCurl.exec();
```

Response

In this example, the response will be in JSON, although you can set the response to be in HTML as well. A sample response based on the example iss shown below:

```
                {
"data": {
  "translations": [
    {
      "translatedText": "Bonjour tout le monde"
      }
    ]
  }
}
```

Close wsCurl

To close the HTTP connection for wsCurl, use the following code:

```
wsCurl.close();
```

Remove References

Your last step will be to remove any references for the garbage collection as shown below:

```
wsCurl=NULL;
```

5. Save your changes and call your new function from your script file.
6. The translated text should be returned to the calling function.

Language Identifiers

Available language identifiers for the wscurl command.

Available language identifiers for the wscurl command.

The following is a list of the available language identifiers that you can use with the wscurl command.

Language Name	Language Code
AFRIKAANS	af
ALBANIAN	sq
AMHARIC	am
ARABIC	ar
ARMENIAN	hy
AZERBAIJANI	az

BASQUE	eu
BELARUSIAN	be
BENGALI	bn
BIHARI	bh
BULGARIAN	bg
BURMESE	my
CATALAN	ca
CHEROKEE	chr
CHINESE	zh
CHINESE_SIMPLIFIED	zh-CN
CHINESE_TRADITIONAL	zh-TW
CROATIAN	hr
CZECH	cs
DANISH	da
DHIVEHI	dv
DUTCH	nl
ENGLISH	en
ESPERANTO	eo
ESTONIAN	et
FILIPINO	tl
FINNISH	fi
FRENCH	fr
GALICIAN	gl
GEORGIAN	ka
GERMAN	de
GREEK	el
GUARANI	gn
GUJARATI	gu
HEBREW	iw
HINDI	hi
HUNGARIAN	hu
ICELANDIC	is

INDONESIAN	id
INUKTITUT	iu
ITALIAN	it
JAPANESE	ja
KANNADA	kn
KAZAKH	kk
KHMER	km
KOREAN	ko
KURDISH	ku
KYRGYZ	ky
LAOTHIAN	lo
LATVIAN	lv
LITHUANIAN	lt
MACEDONIAN	mk
MALAY	ms
MALAYALAM	ml
MALTESE	mt
MARATHI	mr
MONGOLIAN	mn
NEPALI	ne
NORWEGIAN	no
ORIYA	or
PASHTO	ps
PERSIAN	fa
POLISH	pl
PORTUGUESE	pt-PT
PUNJABI	pa
ROMANIAN	ro
RUSSIAN	ru
SANSKRIT	sa
SERBIAN	sr
SINDHI	sd

SINHALESE	si
SLOVAK	sk
SLOVENIAN	sl
SPANISH	es
SWAHILI	sw
SWEDISH	sv
TAJIK	tg
TAMIL	ta
TAGALOG	tl
TELAGU	te
THAI	th
TIBETAN	bo
TURKISH	tr
UKRANIAN	uk
URDU	ur
UZBEK	uz
UIGHUR	ug
VIETNAMESE	vi
UNKNOWN	

WsCurl APIs

The following is a list of the available APIs that you can use with the wsCurl command.

- setopt
- exec
- close
- slist_append
- slist_free_all

These are covered in the following sections.

setopt

The setopt API enables you to specify options and parameters for the wsCurl functionality. The syntax is as follows:

```
wsCurl.setopt(Curl.<CURL_OPTIONS>,parameter);
```

Option	Integer Value	Definition
CURLOPT_URL	10002	This is the name of the proxy, if a proxy is used. The second parameter should be a string.
CURLOPT_PROXY	10004	This is the full URL that you will perform either a get or a put action upon. The second parameter should be a string.
CURLOPT_PROXYUSRPWD	10006	This is the name and password combination you will use when fetching. The second parameter should be a string.
CURLOPT_PROXYPORT	59	This is the proxy port that you will use. The second parameter should be an integer.
CURLOPT_TIMEOUT	13	This is the number of seconds before the read operation times out. The second parameter should be an integer.
CURLOPT_HEADER	42	This is a Boolean defining if the header will be thrown out in response to the exec operation. The second parameter should be either '0' or '1'. '0' specifies that the header will not be included; '1' specifies that the header will be included.
CURLOPT_SSL_VERIFYPEER	64	This parameter specifies if the verify peer operation in the SSL handshake will or will not occur. . The second parameter should be either '0' or '1'. '0' specifies that the peer will not be verified; '1' specifies that the peer will be verified.
CURLOPT_SSL_VERIFYHOST	81	This parameter specifies if the verify host operation in the SSL handshake will or will not occur. . The second parameter should be

		either '0', '1', or '2'. '0' specifies that the host will not be verified; '1' specifies that the host's existence will be checked and '2' specifies that the the host matches the provided hostname.
CURLOPT_USERNAME	10173	The email address that will be used to send the mail. The second parameter should be a string.
CURLOPT_PASSWORD	10174	Specifies the password associated with the email address we are using to send the mail. The second parameter should be a string.
CURLOPT_MAIL_FROM	10186	Specifies the SMTP mail originator. The second parameter should be a string.
CURLOPT_MAIL_RCPT	10187	Specifies the SMTP mail recipient. The second parameter should be a string.
CURLOPT_VERBOSE	41	Specifies that step-by-step execution will be recorded and turns on logging. the second parameter should be either '0' or '1'. '0' specifies that verbose mode is not on; '1' specifies that verbose mode and logging are on.
CURLOPT_MAXREDIRS	68	Specifies the maximum number of HTTP redirects that will occur. You can use any number up to XX. The second parameter should be an integer.
CURLOPT_READDATA	10009	Specifies the file stream to use as the input. We will use the email header and body as the input stream. The second parameter should be a string.
CURLOPT_USE_SSL	119	Specifies if SSL/TLS for FTP is enabled. There are four options you can choose, which are as follows. The second parameter should be an integer specifying which option you select. • USESSL_NONE: SSL will not be used. The integer value

		is 0. • USESSL_TRY: The connection will attempt to use SSL, but will proceed regardless if SSL can or cannot be used. The integer value is 1. • USESSL_CONTROL: If SSL canot be used, the control connection will fail. The integer value is 2. • USESSL_ALL: If SSL cannot be used, all connections will fail. The integer value is 3.

exec

The exec API enables you to specify a response to the query you are making. The response can be either JSON or HTML and the syntax is as follows:

```
var response = wsCurl.exec();
```

You can use the following syntax to check for the response:

```
println(wsCurl.error);
```

The exec API contains a number of response codes that may be returned. As defined by the Curl project, the currently available response codes are as follows:

Option	Integer Value	Definition
CURLE_OK	0	No errors occurred - the operation completed normally.
CURLE_UNSUPPORTED_PROTOCOL	1	The target URL is using a protocol used is not supported. Please check if the problem might be one of the following: • Compile-time option that was not used.

		• Misspelled protocol string. • Protocol that libcurl does not have any code for.
CURLE_FAILED_INIT	2	An early initialization failed and the operation did not complete.
CURLE_URL_MALFORMAT	3	The target URL was not properly formatted.
CURLE_NOT_BUILT_IN	4	A required feature, proptocol, or option was found to be missing from your version of libcurl. You will need to get a rebuilt libcurl in order for this to function.
CURLE_COULDNT_RESOLVE_PROXY	5	Could not resolve the given proxy host.
CURLE_COULDNT_RESOLVE_HOST	6	Could not resolve the given remote host.
CURLE_COULDNT_CONNECT	7	Could not connect to either the host or the proxy.
CURLE_FTP_WEIRD_SERVER_RESPONSE	8	Unexpected or improper response received from FTP server. This may mean that the given server is not a valid FTP server.
CURLE_REMOTE_ACCESS_DENIED	9	The request for a remote connection to the given resource was denied.
CURLE_FTP_ACCEPT_FAILED	10	FTP handshake did not succeed and the operation did not complete.
CURLE_FTP_WEIRD_PASS_REPLY	11	Unexpected or improper response received after sending FTP password to server.
CURLE_FTP_ACCEPT_TIMEOUT	12	Timeout occurred during an active FTP session.
CURLE_WEIRD_PASV_REPLY	13	Unexpected or improper response received in response to either a PASV or a EPSV command.

CURLE_FTP_WEIRD_227_HOST	14	FTP servers typically return a 227-line in response to a PASV command. This code is returned if the libcurl fails to parse that response.
CURLE_FTP_CANT_GET_HOST	15	Internal failure occurred while looking up the given host for a new connection.
CURLE_FTP_COULDNT_SET_TYPE	17	Error occurred while tyring to set the transfer mode to either ASCII or binary.
CURLE_PARTIAL_FILE	18	The file transfer was either shorter or longer than expected. This typically occurs when the server first reports a transfer size and then the actual size fails to match the reported size.
CURLE_FTP_COULDNT_RETR_FILE	19	Unexpected or improper response received in response to a RETR command. This error can also occur when a a zero byte transfer completes.
CURLE_QUOTE_ERROR	21	This error typically occurs when a 400 or higher error code is returned from the FTP server while a custom QUOTE command is being sent.
CURLE_HTTP_RETURNED_ERROR	22	This error is returned if the CURLOPT_FAILONERROR option is set to TRUE and the HTTP server returns an error code that is 400 or higher.
CURLE_WRITE_ERROR	23	This code is returned either when an error occurs while writing data to a local file or when an error is returned from a write callback.
CURLE_UPLOAD_FAILED	25	There was a failure starting the upload. For FTP, this typically occurs when the server denies the STOR comand. Please see the error buffer fto find why the STOR command was denied.

CURLE_READ_ERROR	26	This code is returned either when an error occurs while reading a local file or when an error is returned from a read callback.
CURLE_OUT_OF_MEMORY	27	This code will be returned when a memory allocation request fails. Warning: This code indicates a very serious problem is occurring. However, this code can sometimes indicate a conversion error intead of a memory allocation error if the CURL_DOES_CONVERSIONS option is defined.
CURLE_OPERATION_TIMEOUT	28	The specified timeout was reached and the operation was terminated.
CURLE_FTP_PORT_FAILED	30	The FTP PORT command returned an error. This typically occurs if you specified an improper or incomplete address.
CURLE_FTP_COULDNT_USE_REST	31	An error was returned from the FTP REST command. Note: There is a problem with the server - it should never occur if the server is OK.
CURLE_RANGE_ERROR	33	The RANGE command did not work. This typically occurs if the server does not support range requests.
CURLE_HTTP_POST_ERROR	34	This error usually occurs due to some kind of internal confusion. Check your settings and your given connection parameters.
CURLE_SSL_CONNECT_ERROR	35	A problem has occurred in the SSL/TLS handshake. Some possible cause are certificate format, path, or permission errors. Password errors may also be the reason, as well as

		other unrelated errors. Consult the error buffer for a more specific explanation of the error.
CURLE_BAD_DOWNLOAD_RESUME	36	The download could not be resumed because the specified offset was out of the file boundary.
CURLE_FILE_COULDNT_READ_FILE	37	THis means that a given file could not be opened. This error typically occurs when the given file path does not identify a specific file. Check file permissions as well - they also may cause this error code to be returned.
CURLE_LDAP_CANNOT_BIND	38	LDAP cannot bind - the bind operation failed.
CURLE_LDAP_SEARCH_FAILED	39	The LDAP search operation failed.
CURLE_FUNCTION_NOT_FOUND	41	A required zlib function was not found.
CURLE_ABORTED_BY_CALLBACK	42	A callback returned the 'abort' command.
CURLE_BAD_FUNCTION_ARGUMENT	43	This means an internal error has occurred. Usually this is due to a function being called with a bad parameter.
CURLE_INTERFACE_FAILED	45	An interface error has occurred. Typically, this occurs when the outgoing interface has not been correctly set. You can use the CURLOPT_INTERFACE option to specify the correct interface's outgoing IP address.
CURLE_TOO_MANY_REDIRECTS	47	Too many redirects have occurred or the maximum number of redirects haS been reached. You can use the CURLOPT_MAXREDIRS option to set the number of redirects that are allowed.
CURLE_UNKNOWN_OPTION	48	An option that was passed to libcurl is either not known or

		not recognized. You will need to consult the appropriate documentation but this is typically a problem in whaever application is using libcurl. Consult the error buffer, as it may contain more detailed information about the precise issue.
CURLE_TELNET_OPTION_SYNTAX	49	A telnet option string was illegally or incorrectly formatted.
CURLE_PEER_FAILED_VERIFICATION	51	The server's SSL certificate of SSH md5 fingerprint is not correct.
CURLE_GOT_NOTHING	52	No response was received from the server.
CURLE_SSL_ENGINE_NOTFOUND	53	TCOuld not find the specified SSL crypto engine.
CURLE_SSL_ENGINE_SETFAILED	54	Could not set the specified SSL crypto engine as the default.
CURLE_SEND_ERROR	55	Could not send network data - the attempt failed.
CURLE_RECV_ERROR	56	Could not receive network data - the attempt failed.
CURLE_SSL_CERTPROBLEM	58	There is a problem with the local client certificate.
CURLE_SSL_CIPHER	59	Could not use the specified SSL cypher.
CURLE_SSL_CACERT	60	Could not use any known CA certificates to authenticate peer certificate.
CURLE_BAD_CONTENT_ENCODING	61	The specified transfer encoding is not recognized.
CURLE_LDAP_INVALID_URL	62	The specified LDAP URL is invalid.
CURLE_FILESIZE_EXCEEDED	63	The file exceeds the maximum size limits.
CURLE_USE_SSL_FAILED	64	The requested SSL FTP level failed.
CURLE_SEND_FAIL_REWIND	65	Attempt to rewind data during

		a send operation failed.
CURLE_SSL_ENGINE_INITFAILED	66	Could not initialize the SSL engine - the attempt failed.
CURLE_LOGIN_DENIED	67	The remote server denied login permission.
CURLE_TFTP_NOTFOUND	68	The specified file is not found on the TFTP server.
CURLE_TFTP_PERM	69	There is a permission problem on the TFTP server.
CURLE_REMOTE_DISK_FULL	70	The server has run out of disk space.
CURLE_TFTP_ILLEGAL	71	The attempted TFTP operation is illegal.
CURLE_TFTP_UNKNOWNID	72	The specified TFTP transfer ID is unknown.
CURLE_REMOTE_FILE_EXISTS	73	The specified file already exists and will not be over-written.
CURLE_TFTP_NOSUCHUSER	74	This error indicates that there is a problem with the server - a TFTP server that is working correctly should never return this error.
CURLE_CONV_FAILED	75	Character conversion failed.
CURLE_CONV_REQD	76	Caller must register coversion callbacks in order for the operation to succeed.
CURLE_SSL_CACERT_BADFILE	77	A problem occurred while reading the SSL CA certificate path. This could be due to access rights or an incorrect path.
CURLE_REMOTE_FILE_NOT_FOUND	78	The resource specified in the URL does not exist.
CURLE_SSH	79	
CURLE_SSL_SHUTDOWN_FAILED	80	The SSL connection failed to shut down.
CURLE_AGAIN	81	This means that the SSL socket is not ready to send or receive. You must wait until it is ready and try again.

		Note: This code is only returned from curl_easy_recv and curl_easy_send.
CURLE_SSL_CRL_BADFILE	82	The CRL file could not be loaded.
CURLE_SSL_ISSUER_ERROR	83	The SSL issuer check failed.
CURLE_FTP_PRET_FAILED	84	This error usually occurs either when the FTP server does not understand the PRET command or when it does not support the given argument.
CURLE_RTSP_CSEQ_ERROR	85	The RTSP CSeq numbers do not match.
CURLE_RTSP_SESSION_ERROR	86	The RTSP CSeq session identifiers do not match.
CURLE_FTP_BAD_FILE_LIST	87	Could not parse the FTP file list. This usually occurs during FTP wildcard loading.
CURLE_CHUNK_FAILED	88	An error occurred during the chunk callback.

Other APIs

In addition to the Setopt and Exec APIs previously introduced, there are three other APIs that are available in wsCurl. These are as follows:

- close
- slist_append
- slist_free_all

CLOSE

The close API closes the handle to fetch the request and also frees up all memory allocated to the operation. The syntax is as follows:

```
wsCurl.close();
```

There are no options or other parameters associated with the CLOSE API.

SLIST_APPEND

The slist_append API enables you to add email addresses to the list. You can specify if the appended mail is to be on the main recipient list or if it is to be a CC or a BCC. The syntax is as follows:

```
wsCurl.slist_append("<recipient_email@company.com>");
```

Note: Make sure to use the same email addresses as those you defined in the setopt READDATA email header.

There are no options or other parameters associated with the SLIST_APPEND API.

SLIST_FREE_ALL

The slist_free_all API frees all resources allocated to the creation of the eamil lists done through the slist_append API. The syntax is as follows:

```
wsCurl.slist_free_all();
```

There are no options or other parameters associated with the SLIST_APPEND API.

Usage Details

- **WSCurl SimpleHTTP**

  Execute a simple URL and retrieve metadata information using WSCurl.

  ```
  wsCurl.setopt(Curl.CURLOPT_URL, completeURL);
  ```

  Learn more about Retrieve metadata information of a website.

- **WSCurl Get IP Address**

  Retrieve the global IP Address using WSCurl.

  ```
  pushbutton([TOOLBAR], '
  Get IP Address', '?',{"process":getIPAddress});
  ```

  Learn more about WSCurl and how to retrieve the IP address.

- **WSCurl Send Mail**

  Send email using WSCurl from SAP.

```
wsCurl.setopt(Curl.CURLOPT_URL, "smtp://mail.guixt.com:25");
```

Learn More about WSCurl Send Mail functionality.

- **WSCurl Translate Language**

  Using WSCurl, input text in a non-Western language and then use a service such as Google Translate to change the text into English or some other language.

```
pushbutton([TOOLBAR
], 'toTranslate', '?',{
"process":translate,"using":{"toTranslate":g_source1,"targetLang
":g_langugage1}});
```

Learn how WSCurl Translate Language.

Unique solution ID: #1100
Author: Punil Shah
Last update: 2018-07-06 14:20