## Description of the zOS Rexx framework for CSM Rest API

## Table of content

-----------------------------------------------------------------------------------------------------------------------------------

-----------------------------------------------------------------------------------------------------------------------------------

## About the framework

The z/OS Rexx framework for CSM Rest API was developed to demonstrate the z/OS TSO Web Enablement Toolkit capabilities in a simplified manner for utilizing the IBM Copy Services Manager Rest API interface. It enables z System Programmers and Storage Administrators to interact with IBM Copy Services Manager from a z platform without installing the CSM CLI for z/OS.

The z/OS TSO Web Enablement Toolkit is available with z/OS 2.2 or later and provides HTTP and JSON services that are callable through various program languages. For more information, please refer to:

- https://www.ibm.com/docs/en/zos/2.2.0?topic=consider-zos-client-web-enablement-toolkit

The Rexx framework provides functions and procedures to utilize the web enablement toolkit services in Rexx. It re-uses Rexx examples from following sources:

- https://github.com/IBM/zOS-Client-Web-Enablement-Toolkit
- SYS1.SAMPLIB(HWTJSPRT) => Rexx template to print formatted JSON text

It does not only provide necessary HTTP request handling and JSON response parsing, but also other additional features that greatly simplify its usage for CSM Rest API. For usage description of the CSM Rest API, please refer to:

- https://www.ibm.com/docs/en/csm/6.3.1?topic=reference-csm-rest-api-documentation

## Features supported by the Rexx framework for CSM Rest API

1. The Rexx framework supports automated management of CSM server credentials (and http request tokens) encrypted in flat text files or datasets. Each framework user can utilize its own credentials file, which is a customizable execution parameter. If no or invalid saved credentials are found, the user will be prompted for actual CSM server credentials.
   Note: Depending on the execution environment of the Rexx, the password prompt may occur with echo:
   - If running in OMVS shell or ISPF environments, the password prompt will be masked.
   - If the credentials will be prompted in an ISPF environment, it will be done via a dynamic ISPF panel popup. The panel member (and dataset) will be automatically created if not existing. This allows to prompt the password without display.
   - When running the Rexx in plain TSO environments, the password prompt cannot be masked.
   - If the credentials prompt is aborted or no valid credentials are provided, an empty credential template file is created. You can specify valid CSM user credentials in the file directly and they will be encrypted during next Rexx execution using the same credentials file.
2. The http request wrapper function automatically manages all web enablement toolkit tasks required to issue an HTTP request to a CSM server. All requests are using the recommended token-based authentication method. If no token is available or the last token is expired, the wrapper function will request a new token from the CSM server with the provided CSM server credentials.
3. Existing JSON functions of the framework can be utilized to either print a JSON formatted output of the response data, or to parse the JSON text for specific entries and values
4. An optional output USS file can be specified to save the http response data. This is required if you request to download a backup from the CSM server, which will result in a binary octet-

stream response and binary stream data cannot be further parsed or displayed through the JSON functions.

5. The Rexx framework is parameterized to a large extend and supports various execution modes:
   - Executable directly from ISPF or TSO panels
   - Executable from TSO shell
   - Executable from OMVS shell (and as such, also via external ssh calls to OMVS if remote ssh login is configured)
   - Executable via a z/OS job
6. Static parameters for your environment can be hard coded in the Rexx. Other parameters used more dynamically should be specified as execution parameters, which will overwrite the hard coded parameter (default) settings.
7. The default execution mode of the framework is to issue the specified or hardcoded http request and display the JSON formatted response. When using the execution parameter **-u**, you can also dynamically specify and run any defined and allowed framework CSM_ function of the Rexx. There are a couple of example CSM_ functions contained in the Rexx to demonstrate usage of the functions and procedures and how CSM response data can be parsed and displayed. Examples include display of a CSM session, system or scheduled task overview, or to issue a CSM session command or scheduled task command.
8. The main routine, as well as other functions of the Rexx framework can be modified or expanded as required for your needs. Some examples are provided in the main function. The Rexx is provided as is without any warranty or support.

## Limitations and considerations of the Rexx framework

1. The Rexx framework contains code to support the SSL key type option with PKCS11 Tokens or keyrings provided via the Security Facility (e.g. RACF). Although these options can be configured via execution parameter (*-k*) or variable (**g.cKeyRing**), their functionality was not fully tested. The SSL key type option with a PKCS12 key database (*-k* or **g.cKeyDb**) and password stash file (*-s* or **g.cDbStash**) is the tested option.
2. The Rexx framework contains code to support Basic Authentication setup for HTTP requests. This authentication mode however is not utilized for CSM server Rest API because **the preferred token based authentication method is automatically configured** and managed through the HTTP request wrapper function.
3. The HTTP response body is **by default translated from ASCII to EBCEDIC (*A2E*)** to allow parsing of JSON text on the z platform. If stream data is expected in the HTTP response, the body translation of the HTTP request handle needs to be switched off first, before issuing the request. Otherwise an output file receiving the binary data may not be usable. The http_request() wrapper function of the framework automatically disables A2E translation of the response body if it finds a '/download' pattern in the request path. You can also force disabling the A2E translation via a wrapper function parameter if necessary.
4. The **size of the data that can be received by the Rexx framework is limited to 16 MB**. This may be sufficient to download CSM backup files, but not for CSM PePackages. This limitation is based on Rexx variable limits, which are used buffer the response data received by the web enablement toolkit. There is no work around in Rexx to receive unlimited streams as supported by the web enablement toolkit when using non-Rexx implementations supporting program exits.

## Copyright information and disclaimer

This framework is provided for tutorial purposes only. A complete handling of error conditions has not been shown or attempted, and this program has not been submitted to formal IBM testing. This program is distributed on an 'AS IS' basis without any warranties either expressed or implied.

## Install the Rexx framework

Obtain the proper code distribution from github for your preferred usage mode.

**Note:**

The framework for TSO usage and OMVS usage is identical, except of very few default file name settings.

### Installation for TSO usage

Please obtain following file from github and upload it to the Z platform with **BINARY** mode.

CSM.RXCSMAPI.XMT

When allocating a dataset for the upload, use a sequential dataset with **Fixed Block, LRECL 80, Space 1 CYL** and select a proper high level qualifier for the upload dataset, e.g.

```
CSM.RXCSMAPI.XMT     → #hlq.CSM.RESTAPI.XMT
```

After upload, receive the XMIT dataset into a new or existing target DSNs of format PDS(E), RECFM=FB, LRECL=80, SPACE=(CYL,(1,1)), using a proper #hlq, e.g. DSN('#hlq.CSM.RESTAPI'). TSO command example:

```
RECEIVE INDSN('#hlq.CSM.RESTAPI.XMT') DSN('#hlq.CSM.RESTAPI')
```

Resulting dataset content after receive:

**#hlq.CSM.RESTAPI:**

```
CSMAPIJC     → Job template to execute the framework with parameters
ISPFPASS     → ISPF panel definition to prompt for the CSM server credentials
             (member will be created by the framework if not existing but
             required, the default member name can be changed in framework
             settings)
README       → ReadMe documentation
RXCSMAPI     → Executable Rexx member with framework for CSM Rest API.
```

### Installation for OMVS usage

Please obtain following file from github and upload it as USS file to OMVS in **ASCII mode**, e.g. via ftp(s) text transfer to your home folder.

rxcsmapi.rexx

```
rxcsmapi.rexx        → /u/username/csmapi/rxcsmapi.rexx
```

Make sure that the USS file has proper read, write and execution permissions. You can modify that with chmod as required.

### Transfer Rexx Framework between OMVS and TSO

There are various methods to copy files between TSO and OMVS. One easy approach is to use the OMVS **cp** command. For example, to copy the Rexx framework file from a dataset member to your local OMVS path use following command:

```
cp "//'#hlq.CSM.RESTAPI(RXCSMAPI)'" ./rxcsmapi.rexx
```

## Prepare usage of the Rexx framework

### Create the PKCS12 keystore file with public CSM server https certificate

For https communication with the z/OS web enablement toolkit, you need a public certificate that is either configured in a keyring or PKCS11 token in your z/OS SAF (e.g. RACF), or provided in a PKCS12 keystore file (USS file) combined with a V1 stash file (USS file containing the stashed password to access the PKCS12 keystore).

**Note:**

While keystore tools may allow to create a V2 stashed password file, this format seems to cause problems for the z/OS web enablement toolkit to access the PKCS12 keystore.

The public CSM server https certificate required for either the keyring or PKCS12 keystore can be exported from your internet browser. Use a binary DER file format when you export the certificate.

**Note:**

While the framework also supports using a keyring or PKCS11 token, the process to import the certificate into SAF and creation of a keyring or token is not described here. Please refer to your SAF documentation to import certificates and create keyrings or PKCS11 tokens.

Following procedure describes how you can export the public CSM server https certificate and import it into a created PKCS12 keystore file:

1. Export the CSM https certificate from a browser
   - Open the CSM GUI in your browser to see the login page and make sure the server certificate was imported in the browser
   - In your browser, go to the Certificate database (e.g. in Firefox, go to Tools -> Security -> Certificates -> Show Certificates)
   - Select the CSM server certificate and export it. You don't need to export the full certificate chain, only the leaf certificate is sufficient. Make sure to export the certificate as DER file. This is a binary X509 format which can be imported into a PKCS12 keystore file using the IBM Java ikey utility
   - If you can export the certificate only as PEM key file (ASCII file), you must first convert the PEM keyfile to a PKCS12 or DER keyfile. This is possible with openssl tools:
     - Example how to convert a PEM file to a PKCS12 file:
       `openssl pkcs12 -in cert.pem -out cert.p12 -export`
       Enter the export password and the PKCS12 file (pkcs12_file) will be generated.
     - Example how to convert a PEM file to a DER file:
       `openssl x509 -in cert.pem -out cert.der -outform DER`
2. Create a PKCS12 keystore with a V1 stash file containing the password
   - On a system with an IBM Java runtime, you can use the **ikeyman** graphical tool or the **ikeycmd** command line tool to create the PKCS12 keystore and stashfile. The graphical ikeyman tool is not explained in more detail at his point. If you don't have a local IBM Java runtime, use the IBM Java version in OMVS. There you can use the command line tool ikeycmd as described in following steps. Make sure that your Java runtime binaries are accessible through your user shell. You can verify this with the shell commands:
     - `java -version`
     - `ikeycmd -?`
   - Create the PKCS12 keystore with following command:
     ```
     ikeycmd -keydb -create -db csmcerts.p12 -type pkcs12 -pw password -stash -
     v1stash
     ```

- Define your keystore db file name and password as required. The stash file name will be the keystore db file name with an .sth extension. The command will create two files in the local directory:
  - *csmcerts.p12*
  - *csmcerts.sth*

3. Import the CSM https certificate into the keystore
   - Transfer the exported DER certificate file to the system and path where you created the PKCS12 keystore db. If you transfer it from your PC to OMVS, make sure to use a binary transfer mode.
   - Import the certificate with following command:
     ```
     ikeycmd -cert -add -file csmserver1.der -format binary -label csmserver1 -
     trust enable -db csmcerts.p12 -type pkcs12 -stashed
     ```
   - Specify the DER file name with the certificate to import and select an appropriate label for the certificate. You need to reference the label later in the framework.
   - If you have more CSM servers you want to connect to, do the whole process for the other CSM servers as well. You can import the certificates into the same keystore with different labels. This allows you to skip creation of another keystore.
4. List the existing certificates in a keystore
   - To verify or to remember which certificates and labels are contained in the keystore, you can use following commands:
   - List all certificate labels in the keystore:
     ```
     ikeycmd -cert -list -db csmcerts.p12 -stashed
     ```
   - Example:
     Certificates in database
     ```
     /u/username/csmcerts.p12:
         csmserver1
         csmserver2
         csmserverzos
     ```
   - List certificate details:
     ```
     ikeycmd -cert -details -label csmserver1 -db csmcerts.p12 -stashed
     ```
5. If you created the PKCS12 keystore on your local PC, upload it to an OMVS file in binary mode. This makes it usable for the Rexx framework as required for the z/OS web enablement toolkit.
6. Verify the OMVS access permissions to the keystore and stash files to ensure they can be read by required users or groups. You should restrict read access to necessary users and groups only. Modifications can be performed with the `chmod` command.

## Required and optional file definitions for the execution

You need to define or provide some DSN(member) or USS filenames for configuration files as required by the framework execution. These names can be hard coded in the variable initialization function as customized framework defaults (see **InitializeVars:**), or specified as parameters for each framework execution. A specified execution parameter overwrites the hard coded variable initialization settings.

Following file types are used when the Rexx framework is executed, **required file definitions are bold**. The corresponding global variable name as well as the execution parameter are listed for each file):

- **USS Keystore file with SSL certificate** for the CSM server (**g.cKeyDb** or **-k** parameter)
  Must be a binary USS file in PKCS12 format containing one or more certificates for the https server connection encryption. Access to this file should be limited to allowed users or groups only

- **USS file with stashed password to access the keystore file** (**g.cDbStash** or **-s** parameter)
  Must be a binary USS file in PKCS12 stashed V1 format to access the corresponding keystore. Access to this file should be limited to allowed users or groups only
- **DSN/file to save encrypted server credentials** (**g.AuthFile** or **-c** parameter)
  Can be a sequential dataset, or dataset member or USS file. Access to this dataset or file should be limited to allowed users or groups only
- **DSN/file to save encryption private key** (**g.EncrFile** or **-e** parameter)
  Can be a sequential dataset, or dataset member or USS file. Access to this dataset or file should be limited to allowed users or groups only. It will contain the generated private key to encrypt the credentials and access token in the server credentials file
- USS output file to save http response data (**g.OutFile** or **-o** parameter)
  If specified, the http response will be saved in that file. Therefore the output file will be managed only as binary USS file to fully support received binary stream data without any conversion. If JSON text is received, it will contain the unformatted JSON response from the server. The content of the output file is overwritten by each http response. To keep multiple responses, you can specify different output files for each request.
- DSN Member for ISPF credentials prompt panel (**g.IPanDsn**)
  Must be a dataset member and is required only when credentials are prompted while executing in an ISPF environment (This file definition is not provided as execution parameter setting)
- DSN/file for HTTP connection verbose trace, default = STDOUT (**g.TraceFile** or **-t** parameter)
  This is optional and may be required only for debugging HTTP web enablement toolkit handling. It will be used only when the Verbose option is enabled.

In general, the framework can run in a TSO environment but use USS files and vice versa. However, some of the used files require a specific format, e.g. the PKCS12 keystore files or the optional HTTP response output file must be USS files, while the ISPF authentication panel must be a dataset member. All defined datasets, members or USS files in the framework are automatically allocated if not existing. If only a member name is defined, e.g. ’(*mbrname*)’, the framework will consider the Rexx execution dataset as dataset location for the member. This will fail, if the Rexx framework is executed from a USS file or a sequential dataset instead of a member. In that case, define the full ‘*dataset*(*member*)’ names in the file definition. You can also utilize a common prefix variable (**g.Pref**), which can be used to define a common PDS/PDSE library or USS path for your framework configuration files.

# How to use the Rexx framework:

## Rexx framework execution parameters:

Command line execution parameters:

```
-h: Host URI with protocol, host, port to be used for the connection
-k: Key database file or keyring with certificate for HTTPS connections
-s: Stash file to access the key database file
-l: Label of certificate in PKCS12 key database
-u: Use specified internal CSM function (will ignore -r -p -d -f)
-r: Request type: GET, PUT, POST, DELETE, HEAD
-p: Full URI path to the requested service
-d: Data to send in request body, such as input parameter
-c: USS file or DSN(Mbr) to save CSM server credentials
-e: USS file or DSN(Mbr) with encryption key for server credentials
-i: Enable informative output (Default is disabled)
-v: Enable verbose output (Default is disabled)
-t: Optional Trace File for verbose connection output (Default Stdout)
-f: Filter for JSON root object entries to be displayed
-o: USS Output file to save response data (Required for Stream data)
```

Example:

```
./rxcsmapi.rexx -h "https://hostname:port" -k "/u/username/keystore.p12" -s
"/u/username/keystore.sth" -l "certlabel" -c "/u/username/cred.txt" -e
"/u/username/cred.key" -r "POST" -p
"/CSM/web/sessions/<name>/backups/H1/<backupid>" -d "cmd=Recover%20Backup" -i
-f "msgTranslated","timestamp"
```

Valid functions for the **-u** parameter (Mandatory parameters are **bold**):

```
CSM_SessOverview(hdr,fmt,delim,sort)
CSM_SysOverview(hdr,fmt,delim,sort)
CSM_PathOverview(hdr,fmt,delim,sort)
CSM_TaskOverview(hdr,fmt,delim,sort)
CSM_GetSysPaths(sys,hdr,fmt,delim,sort)
CSM_GetSessCpSets(sess,cols,hdr,fmt,delim,sort)
CSM_GetSessBackups(sess,hdr,fmt,delim,sort)
CSM_GetSessCmd(sess,hdr,fmt,delim,sort)
CSM_RunSessCmd(sess,cmd,parm)
CSM_RunHaCmd(cmd,server:port,user,pwd)
CSM_RunTaskCmd(taskid,cmd,datetime,sync)
CSM_ShowTask(task,hdr,fmt,delim)
```

*Hdr*, *fmt*, *delim* and *sort* are optional parameters to format the table display of the internal CSM function. To skip a parameter position in the declaration, just leave it blank. Trailing parameters that are not needed can be skipped completely. *Hdr* will display header information for the function and the table columns if enabled (default). *Fmt* will align the table fields with spaces if enabled (default). Left alignment will be used unless the column name starts with # (indicating number values), which will be right aligned. *Hdr* and *fmt* can be **0** or **OFF** to disable the option, and **1** or **ON** to enable the option. *Delim* is a field separator character. If you want a blank or comma as separator, the parameter must be quoted. If any of those parameters is blank or not specified, the default hard coded settings are used (**g.tHeader**=1, **g.tFormat**=1, **g.tDelim**='|').

The *sort* parameter must be a comma separated list of column number with an optional sort direction character **A** or **D** (default is **A**scending). For example “4,2d” will first sort column 4 ascending, and column 2 descending as second order. The default sort setting is unsorted, but each CSM_ function might have implemented its own default sort setting to display the table rows in a meaningful order.

For example, `CSM_SessOverview(,OFF,’,’)` will display the output with default header setting, no table formatting, comma as field separator and using default sort options of the function.

*Sys, sess*, *cmd*, *task*, *taskid* are mandatory parameters in the corresponding functions. If blanks are used in session names or commands, the parameters must be quoted to recognize them as single parameter.

For other function parameters not explained, please refer to the corresponding CSM_ function description to understand their purpose and required format.

The Rexx framework argument parsing routine for the -u parameter setting will ensure to quote all function parameters that are not numerical or blank, in order to ensure they are properly interpreted for the function call. If you need to add additionally created internal functions to the -u parameter, you need to add the new function name to the defined valid functions in the argument parsing function **ParseArgs**.

## Job template

Following is a job template (JCL) to execute the Rexx framework in TSO with parameters. The template assumes the Rexx framework is a member (RXCSMAPI) in the defined EXEDSN.

```
//Jobcard
//* Define the dataset of the Rexx executable
//*----------------------------------------------------
// SET  EXEDSN=#HLQ.CSM.RESTAPI
//*----------------------------------------------------
//* Execute the REXX exec under a TSO environment
//RUN      EXEC PGM=IKJEFT01
//SYSEXEC  DD DSN=&EXEDSN,DISP=SHR
//SYSOUT   DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//* Rexx execution with parameters, line concatenation with -
//SYSTSIN  DD *
 %RXCSMAPI                                                 -
 -h https://csmserver:9559                                 -
 -k /u/tluther/csmcerts.p12                                -
 -s /u/tluther/csmcerts.sth                                -
 -l csmserver1                                             -
 -u CSM_SessOverview
/*
//INPUT    DD *
/*
//
```

Since the JCL line length is limited, you probably need to split the Rexx execution parameeers across multiple input lines for SYSTSIN. You can concatenate multiple Input lines with a dash ‘-‘ at the end of the line. A more detailed job template is included the Rexx framework installation package for TSO use. (see member CSMAPIJC).

## Runtime examples

Following are some examples how you can use the Rexx framework.

### Run CSM Rest API query from OMVS shell and print the formatted JSON response:

```
TLUTHER:/MCECEBC/u/tluther:> ./rxcsmapi.rexx -h https://csmserver:9559
-k /u/tluther/csmcerts.p12 -s /u/tluther/csmcerts.sth -l csmserver1
-c /u/tluther/cred.txt -e /u/tluther/cred.key -r "GET" -p
"/CSM/web/system/ha"
{
  "msg"              : "IWNR3048I",
  "resultText"       : "IWNR3048I [Dec 7, 2021 4:38:26 PM] The high
availability status from server WINDOWS-PJ1KTM6 was successfully queried.",
  "islocalactive"    : true,
  "maxsupportedconnections": 1,
  "localhaport"      : 9561,
  "serverinfo"       : [],
  "inserts"          : [
    "WINDOWS-PJ1KTM6"
  ],
  "timestamp"        : 1638891506538
}
```

### Run job to query CSM session overview via internal function:

```
//Jobcard
//* Define the dataset of the Rexx executable
//*----------------------------------------------------
// SET  EXEDSN=#HLQ.CSM.RESTAPI
//*----------------------------------------------------
//* Execute the REXX exec under a TSO environment
//RUN      EXEC PGM=IKJEFT01
//SYSEXEC  DD DSN=&EXEDSN,DISP=SHR
//SYSOUT   DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//* Rexx execution with parameters, line concatenation with -
//SYSTSIN  DD *
 %RXCSMAPI                                                  -
 -h https://csmserver:9559                                  -
 -k /u/tluther/csmcerts.p12                                 -
 -s /u/tluther/csmcerts.sth                                 -
 -l csmserver1                                              -
 -u CSM_SessOverview
/*
//INPUT    DD *
/*
//
```

**Note:**
In this example, the default credentials file and the encryption file are hard coded and therefore don’t need to be specified as parameter. The same is possible for the server keystore and stash file and certificate label, if not various CSM servers should be queried.

Job Output (not formatted well due to limited row length in this document):

```
https://csmserver (Query: 0.36 sec.): CSM session overview: 10 sessions
SessName                  |#CpSets|ActHost|Status  |State           |HasError|Recoverable|HS-Status
|SessType
------------A3------------|-------|-------|---D2---|----------------|---D1---|-----------|-------------|-------
----------------------------------------------
ITSO_MGM_ACA91            |      8|H1     |Normal  |Prepared        |false   |true       |off          |Metro
Global Mirror
ITSO_MM_LAH81_H1          |      8|H1     |Normal  |Prepared        |false   |true       |off          |Metro
Mirror Failover\/Failback
ITSO_MT_MM_GM_LAH81_ACA91 |      8|H1     |Normal  |Prepared        |false   |true       |off          |Metro
Mirror - Global Mirror
ITSO_SGC_H1_LAH81         |      8|H1     |Normal  |Protected       |false   |true       |off
|Safeguarded Copy
ITSO_SGC_H3_LAH81         |      8|H1     |Normal  |Protected       |false   |true       |off
|Safeguarded Copy
ITSO_SGC_LAH81            |      8|H1     |Normal  |Protected       |false   |true       |off
|Safeguarded Copy
DS-MM                     |     16|H1     |Inactive|Defined         |false   |false      |off          |Metro
Mirror Failover\/Failback
DS-MM-GMed                |     16|H1     |Inactive|Defined         |false   |false      |off          |Metro
Mirror - Global Mirror w\/ Site 3 Global Mirror
DS-MM-GMp                 |     16|H1     |Inactive|Defined         |false   |false      |off          |Metro
Mirror - Global Mirror w\/ Practice
MM HS                     |    192|H1     |Inactive|Defined         |false   |false      |hs not loaded|Metro
Mirror Failover\/Failback
```

**Note:**
The Ax/Dx symbols in the header separator line show the direction and sort order of table columns printed by the Rexx framework. The table display is customizable with the CSM_ function parameters (hdr,fmt,delim,sort). If they are not specified in the function call, the hard coded default settings will be used for the table display (Header=ON, Formatting=ON, Delim=|). The default sort options vary per CSM_ function to sort lines in most meaningful manner.

## Run TSO command to query available commands for a CSM session:

```
MCECEBC                        ISPF Command Shell
 Enter TSO commands below:

 ===> ex 'TLUTHER.CSM.RESTAPI(RXCSMAPI)' '-h https://csmserver:9559
-l csmserver1 -u CSM_GetSessCmd("DS-MM",OFF)'


Enable Copy to Site 1
Start H1->H2
StartGC H1->H2
***
```

**Note:**
In this example, the default required file definitions are hard coded and therefore don't need to be specified as parameter. The executed DSN(member) may have to be quoted if default HLQ prefixing should be avoided. All execution parameters need to be quoted as well to pass it as single TSO command line argument. If individual parameters need to be quoted as well, use different quote type. The Rexx framework CSM_ function displays the available session commands without header information.

## Run job to issue a command to a CSM session:

```
//Jobcard
//* Define the dataset of the Rexx executable
//*-------------------------------------------------------
```

```
// SET  EXEDSN=#HLQ.CSM.RESTAPI
//*----------------------------------------------------
//* Execute the REXX exec under a TSO environment
//RUN      EXEC PGM=IKJEFT01
//SYSEXEC  DD DSN=&EXEDSN,DISP=SHR
//SYSOUT   DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//* Rexx execution with parameters, line concatenation with -
//SYSTSIN  DD *
 %RXCSMAPI                                                  -
 -h https://csmserver:9559                                  -
 -k /u/tluther/csmcerts.p12                                 -
 -s /u/tluther/csmcerts.sth                                 -
 -l csmserver1                                              -
 -u CSM_RunSessCmd('DS-MM',’Start H1->H2’)
/*
//INPUT    DD *
/*
//
```

**Note:**
In this example, the default credentials file and the encryption file are hard coded and therefore don’t need to be specified as parameter. The same is possible for the server keystore and stash file and certificate label, if not various CSM servers should be queried.

Job Output with disabled header information:

```
https://csmserver: Issuing command 'Start H1->H2' to session 'DS-MM' ...
Response (1.09 sec.): IWNR1026I [Dec 14, 2021 2:09:12 PM] The Start H1->H2
command in the DS-MM session completed.
```

**Note:**
The CSM_RunSessCmd return code will be 0 in case the command response results in an Informational message code. Different return codes will be used when the message code is Warning, Error or Severe. The return code is also reflected in the Job result code. The function will also convert blanks in the command to %20 in order to parse the http request parameter properly.

# CSM function output examples

Following is an output example for each provided CSM function in the framework.

## CSM_SessOverview(hdr,fmt,delim,sort)

Print a session overview with data extracted from **GET /CSM/web/sessions/short** query:

```
TLUTHER:/MCECEBC/u/tluther/csmapi:> ./rxcsmapi.rexx -u CSM_SessOverview
https://csmserver (Query: 0.36 sec.): CSM session overview: 10 sessions
SessName                  |#CpSets|ActHost|Status  |State           |HasError|Recoverable|HS-Status    |SessType
------------A3------------|-------|-------|---D2---|----------------|---D1---|-----------|-------------|-------------------------
----------------------------
ITSO_MGM_ACA91            |      8|H1     |Normal  |Prepared        |false   |true       |off          |Metro Global Mirror
ITSO_MM_LAH81_H1          |      8|H1     |Normal  |Prepared        |false   |true       |off          |Metro Mirror
Failover\/Failback
ITSO_MT_MM_GM_LAH81_ACA91 |      8|H1     |Normal  |Prepared        |false   |true       |off          |Metro Mirror - Global
Mirror
ITSO_SGC_H1_LAH81         |      8|H1     |Normal  |Protected       |false   |true       |off          |Safeguarded Copy
ITSO_SGC_H3_LAH81         |      8|H1     |Normal  |Protected       |false   |true       |off          |Safeguarded Copy
ITSO_SGC_LAH81            |      8|H1     |Normal  |Protected       |false   |true       |off          |Safeguarded Copy
DS-MM                     |     16|H1     |Inactive|Defined         |false   |false      |off          |Metro Mirror
Failover\/Failback
DS-MM-GMed                |     16|H1     |Inactive|Defined         |false   |false      |off          |Metro Mirror - Global
Mirror w\/ Site 3 Global Mirror
DS-MM-GMp                 |     16|H1     |Inactive|Defined         |false   |false      |off          |Metro Mirror - Global
Mirror w\/ Practice
MM HS                     |    192|H1     |Inactive|Defined         |false   |false      |hs not loaded|Metro Mirror
Failover\/Failback
```

## CSM_SysOverview(hdr,fmt,delim,sort)

Print a storage system overview with data extracted from **GET /CSM/web/storagedevices** query:

```
TLUTHER:/MCECEBC/u/tluther/csmapi:> ./rxcsmapi.rexx -u CSM_SysOverview
https://9.155.114.38 (Query: 14.27 sec.): CSM storage device overview: 8 devices
DevName          |DevType|Vendor|#Con|Location         |Serial           |DevID                        |Model
-------A2--------|--A1---|------|----|-----------------|-----------------|-----------------------------|-----------
ACA91            |DS8000 |IBM   |   3|Region_A_Site_2  |ACA91            |DS8000:BOX:2107.ACA91        |2107
BRF71            |DS8000 |IBM   |   1|BRF71            |BRF71            |DS8000:BOX:2107.BRF71        |2107
BRX71            |DS8000 |IBM   |   3|Region_A_Site_1  |BRX71            |DS8000:BOX:2107.BRX71        |2107
LAH81            |DS8000 |IBM   |   3|LAH81            |LAH81            |DS8000:BOX:2107.LAH81        |2107
ZA181            |DS8000 |IBM   |   3|Region_B_Site_3  |ZA181            |DS8000:BOX:2107.ZA181        |2107
SVC_CLUSTER_07_08|SVC    |IBM   |   1|SVC_Cluster_07_08|SVC_Cluster_07_08|SVC:CLUSTER:SVC_CLUSTER_07_08|(not found)
SVC_LAB_NEW      |SVC    |IBM   |   1|SVC_Lab_new      |SVC_Lab_new      |SVC:CLUSTER:SVC_LAB_NEW      |(not found)
SVC_09_21        |SVC    |IBM   |   1|SVC_72_73        |svc_09_21        |SVC:CLUSTER:SVC_09_21        |(not found)
```

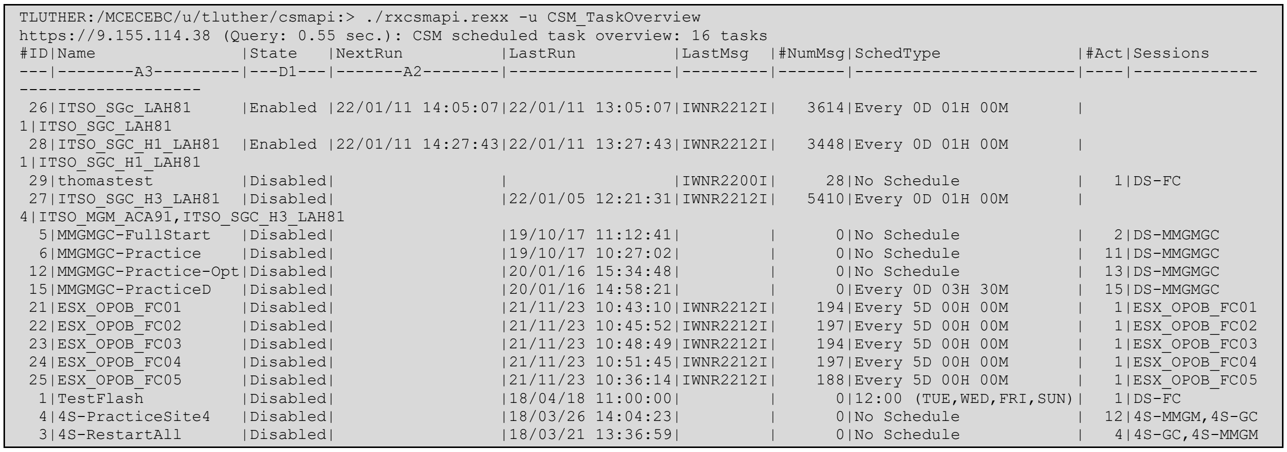
## CSM_PathOverview(hdr,fmt,delim,sort)

Print a PPRC Path overview between DS8000 systems with data extracted from **GET /CSM/web/storagedevices/paths** query and a translated error status:

```
TLUTHER:/MCECEBC/u/tluther/csmapi:> ./rxcsmapi.rexx -u CSM_PathOverview
https://9.155.114.38 (Query: 12.60 sec.): CSM PPRC path overview: 21 System Pairs
SrcSystem |TgtSystem |#GoodPath|#BadPath|ErrStates
----A1----|----A2----|---------|--------|--------------
2107.ACA91|2107.ACA91|      115|       0|
2107.ACA91|2107.BRX71|       41|       0|
2107.ACA91|2107.FAW81|      101|       0|
2107.ACA91|2107.LAH81|       87|       0|
2107.ACA91|2107.ZA181|       87|       0|
2107.BRF71|2107.BRF71|        1|       0|
2107.BRX71|2107.ACA91|       28|       0|
2107.BRX71|2107.BRX71|        3|       0|
2107.BRX71|2107.FAW81|       32|       8|8:FcRetryExd
2107.BRX71|2107.LAH81|        5|       0|
2107.BRX71|2107.ZA181|       26|       0|
2107.LAH81|2107.ACA91|       99|       0|
2107.LAH81|2107.BRX71|       10|       0|
2107.LAH81|2107.FAW81|        2|       0|
2107.LAH81|2107.LAH81|       18|       0|
2107.LAH81|2107.LLB71|        0|      16|16:SecAdptUnav
2107.LAH81|2107.ZA181|       34|       0|
2107.ZA181|2107.ACA91|       84|      10|10:SecSsidMism
2107.ZA181|2107.BRX71|       22|       8|8:SecSsidMism
2107.ZA181|2107.FAW81|       66|       9|9:SecSsidMism
2107.ZA181|2107.LAH81|        4|       0|
```

## CSM_TaskOverview(hdr,fmt,delim,sort)

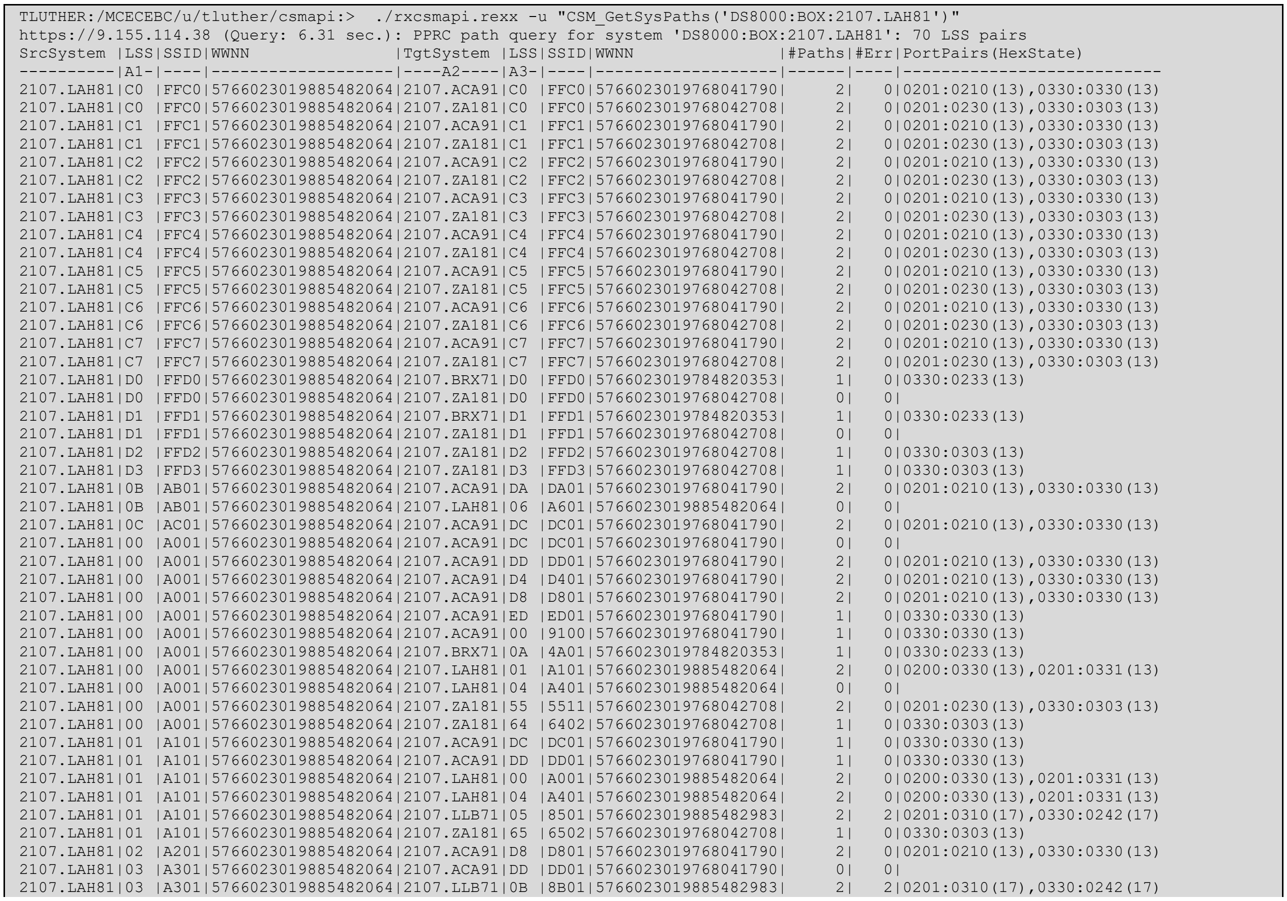
Print a scheduled task overview with data extracted from **GET /CSM/web/sessions/scheduledtasks** query, using a translated schedule definition and local timestamps:



```
TLUTHER:/MCECEBC/u/tluther/csmapi:> ./rxcsmapi.rexx -u CSM_TaskOverview
https://9.155.114.38 (Query: 0.55 sec.): CSM scheduled task overview: 16 tasks
#ID|Name               |State   |NextRun          |LastRun          |LastMsg   |#NumMsg|SchedType               |#Act|Sessions
---|--------A3---------|---D1---|-------A2--------|-----------------|----------|-------|------------------------|----|-------------
-------------------
 26|ITSO_SGc_LAH81     |Enabled |22/01/11 14:05:07|22/01/11 13:05:07|IWNR2212I|   3614|Every 0D 01H 00M        |
1|ITSO_SGC_LAH81
 28|ITSO_SGC_H1_LAH81  |Enabled |22/01/11 14:27:43|22/01/11 13:27:43|IWNR2212I|   3448|Every 0D 01H 00M        |
1|ITSO_SGC_H1_LAH81
 29|thomastest         |Disabled|                 |                 |IWNR2200I|     28|No Schedule             |   1|DS-FC
 27|ITSO_SGC_H3_LAH81  |Disabled|                 |22/01/05 12:21:31|IWNR2212I|   5410|Every 0D 01H 00M        |
4|ITSO_MGM_ACA91,ITSO_SGC_H3_LAH81
  5|MMGMGC-FullStart   |Disabled|                 |19/10/17 11:12:41|          |      0|No Schedule             |   2|DS-MMGMGC
  6|MMGMGC-Practice    |Disabled|                 |19/10/17 10:27:02|          |      0|No Schedule             |  11|DS-MMGMGC
 12|MMGMGC-Practice-Opt|Disabled|                 |20/01/16 15:34:48|          |      0|No Schedule             |  13|DS-MMGMGC
 15|MMGMGC-PracticeD   |Disabled|                 |20/01/16 14:58:21|          |      0|Every 0D 03H 30M        |  15|DS-MMGMGC
 21|ESX_OPOB_FC01      |Disabled|                 |21/11/23 10:43:10|IWNR2212I|    194|Every 5D 00H 00M        |   1|ESX_OPOB_FC01
 22|ESX_OPOB_FC02      |Disabled|                 |21/11/23 10:45:52|IWNR2212I|    197|Every 5D 00H 00M        |   1|ESX_OPOB_FC02
 23|ESX_OPOB_FC03      |Disabled|                 |21/11/23 10:48:49|IWNR2212I|    194|Every 5D 00H 00M        |   1|ESX_OPOB_FC03
 24|ESX_OPOB_FC04      |Disabled|                 |21/11/23 10:51:45|IWNR2212I|    197|Every 5D 00H 00M        |   1|ESX_OPOB_FC04
 25|ESX_OPOB_FC05      |Disabled|                 |21/11/23 10:36:14|IWNR2212I|    188|Every 5D 00H 00M        |   1|ESX_OPOB_FC05
  1|TestFlash          |Disabled|                 |18/04/18 11:00:00|          |      0|12:00 (TUE,WED,FRI,SUN)|   1|DS-FC
  4|4S-PracticeSite4   |Disabled|                 |18/03/26 14:04:23|          |      0|No Schedule             |  12|4S-MMGM,4S-GC
  3|4S-RestartAll      |Disabled|                 |18/03/21 13:36:59|          |      0|No Schedule             |   4|4S-GC,4S-MMGM
```

## CSM_GetSysPaths(sys,hdr,fmt,delim,sort)

Print PPRC Path details for a given DS8000 system with data extracted from **GET /CSM/web/storagedevices/paths/{sys}** query and a hex path status per port pair:



```
TLUTHER:/MCECEBC/u/tluther/csmapi:>  ./rxcsmapi.rexx -u "CSM_GetSysPaths('DS8000:BOX:2107.LAH81')"
https://9.155.114.38 (Query: 6.31 sec.): PPRC path query for system 'DS8000:BOX:2107.LAH81': 70 LSS pairs
SrcSystem |LSS|SSID|WWNN               |TgtSystem |LSS|SSID|WWNN               |#Paths|#Err|PortPairs(HexState)
----------|A1-|----|-------------------|----A2----|A3-|----|-------------------|------|----|--------------------------
2107.LAH81|C0 |FFC0|5766023019885482064|2107.ACA91|C0 |FFC0|5766023019768041790|     2|   0|0201:0210(13),0330:0330(13)
2107.LAH81|C0 |FFC0|5766023019885482064|2107.ZA181|C0 |FFC0|5766023019768042708|     2|   0|0201:0230(13),0330:0303(13)
2107.LAH81|C1 |FFC1|5766023019885482064|2107.ACA91|C1 |FFC1|5766023019768041790|     2|   0|0201:0210(13),0330:0330(13)
2107.LAH81|C1 |FFC1|5766023019885482064|2107.ZA181|C1 |FFC1|5766023019768042708|     2|   0|0201:0230(13),0330:0303(13)
2107.LAH81|C2 |FFC2|5766023019885482064|2107.ACA91|C2 |FFC2|5766023019768041790|     2|   0|0201:0210(13),0330:0330(13)
2107.LAH81|C2 |FFC2|5766023019885482064|2107.ZA181|C2 |FFC2|5766023019768042708|     2|   0|0201:0230(13),0330:0303(13)
2107.LAH81|C3 |FFC3|5766023019885482064|2107.ACA91|C3 |FFC3|5766023019768041790|     2|   0|0201:0210(13),0330:0330(13)
2107.LAH81|C3 |FFC3|5766023019885482064|2107.ZA181|C3 |FFC3|5766023019768042708|     2|   0|0201:0230(13),0330:0303(13)
2107.LAH81|C4 |FFC4|5766023019885482064|2107.ACA91|C4 |FFC4|5766023019768041790|     2|   0|0201:0210(13),0330:0330(13)
2107.LAH81|C4 |FFC4|5766023019885482064|2107.ZA181|C4 |FFC4|5766023019768042708|     2|   0|0201:0230(13),0330:0303(13)
2107.LAH81|C5 |FFC5|5766023019885482064|2107.ACA91|C5 |FFC5|5766023019768041790|     2|   0|0201:0210(13),0330:0330(13)
2107.LAH81|C5 |FFC5|5766023019885482064|2107.ZA181|C5 |FFC5|5766023019768042708|     2|   0|0201:0230(13),0330:0303(13)
2107.LAH81|C6 |FFC6|5766023019885482064|2107.ACA91|C6 |FFC6|5766023019768041790|     2|   0|0201:0210(13),0330:0330(13)
2107.LAH81|C6 |FFC6|5766023019885482064|2107.ZA181|C6 |FFC6|5766023019768042708|     2|   0|0201:0230(13),0330:0303(13)
2107.LAH81|C7 |FFC7|5766023019885482064|2107.ACA91|C7 |FFC7|5766023019768041790|     2|   0|0201:0210(13),0330:0330(13)
2107.LAH81|C7 |FFC7|5766023019885482064|2107.ZA181|C7 |FFC7|5766023019768042708|     2|   0|0201:0230(13),0330:0303(13)
2107.LAH81|D0 |FFD0|5766023019885482064|2107.BRX71|D0 |FFD0|5766023019784820353|     1|   0|0330:0233(13)
2107.LAH81|D0 |FFD0|5766023019885482064|2107.ZA181|D0 |FFD0|5766023019768042708|     0|   0|
2107.LAH81|D1 |FFD1|5766023019885482064|2107.BRX71|D1 |FFD1|5766023019784820353|     1|   0|0330:0233(13)
2107.LAH81|D1 |FFD1|5766023019885482064|2107.ZA181|D1 |FFD1|5766023019768042708|     0|   0|
2107.LAH81|D2 |FFD2|5766023019885482064|2107.ZA181|D2 |FFD2|5766023019768042708|     1|   0|0330:0303(13)
2107.LAH81|D3 |FFD3|5766023019885482064|2107.ZA181|D3 |FFD3|5766023019768042708|     1|   0|0330:0303(13)
2107.LAH81|0B |AB01|5766023019885482064|2107.ACA91|DA |DA01|5766023019768041790|     2|   0|0201:0210(13),0330:0330(13)
2107.LAH81|0B |AB01|5766023019885482064|2107.LAH81|06 |A601|5766023019885482064|     0|   0|
2107.LAH81|0C |AC01|5766023019885482064|2107.ACA91|DC |DC01|5766023019768041790|     2|   0|0201:0210(13),0330:0330(13)
2107.LAH81|00 |A001|5766023019885482064|2107.ACA91|DC |DC01|5766023019768041790|     0|   0|
2107.LAH81|00 |A001|5766023019885482064|2107.ACA91|DD |DD01|5766023019768041790|     2|   0|0201:0210(13),0330:0330(13)
2107.LAH81|00 |A001|5766023019885482064|2107.ACA91|D4 |D401|5766023019768041790|     2|   0|0201:0210(13),0330:0330(13)
2107.LAH81|00 |A001|5766023019885482064|2107.ACA91|D8 |D801|5766023019768041790|     2|   0|0201:0210(13),0330:0330(13)
2107.LAH81|00 |A001|5766023019885482064|2107.ACA91|ED |ED01|5766023019768041790|     1|   0|0330:0330(13)
2107.LAH81|00 |A001|5766023019885482064|2107.ACA91|00 |9100|5766023019768041790|     1|   0|0330:0330(13)
2107.LAH81|00 |A001|5766023019885482064|2107.BRX71|0A |4A01|5766023019784820353|     1|   0|0330:0233(13)
2107.LAH81|00 |A001|5766023019885482064|2107.LAH81|01 |A101|5766023019885482064|     2|   0|0200:0330(13),0201:0331(13)
2107.LAH81|00 |A001|5766023019885482064|2107.LAH81|04 |A401|5766023019885482064|     0|   0|
2107.LAH81|00 |A001|5766023019885482064|2107.ZA181|55 |5511|5766023019768042708|     2|   0|0201:0230(13),0330:0303(13)
2107.LAH81|00 |A001|5766023019885482064|2107.ZA181|64 |6402|5766023019768042708|     1|   0|0330:0303(13)
2107.LAH81|01 |A101|5766023019885482064|2107.ACA91|DC |DC01|5766023019768041790|     1|   0|0330:0330(13)
2107.LAH81|01 |A101|5766023019885482064|2107.ACA91|DD |DD01|5766023019768041790|     1|   0|0330:0330(13)
2107.LAH81|01 |A101|5766023019885482064|2107.LAH81|00 |A001|5766023019885482064|     2|   0|0200:0330(13),0201:0331(13)
2107.LAH81|01 |A101|5766023019885482064|2107.LAH81|04 |A401|5766023019885482064|     2|   0|0200:0330(13),0201:0331(13)
2107.LAH81|01 |A101|5766023019885482064|2107.LLB71|05 |8501|5766023019885482983|     2|   2|0201:0310(17),0330:0242(17)
2107.LAH81|01 |A101|5766023019885482064|2107.ZA181|65 |6502|5766023019768042708|     1|   0|0330:0303(13)
2107.LAH81|02 |A201|5766023019885482064|2107.ACA91|D8 |D801|5766023019768041790|     2|   0|0201:0210(13),0330:0330(13)
2107.LAH81|03 |A301|5766023019885482064|2107.ACA91|DD |DD01|5766023019768041790|     0|   0|
2107.LAH81|03 |A301|5766023019885482064|2107.LLB71|0B |8B01|5766023019885482983|     2|   2|0201:0310(17),0330:0242(17)
```

```
2107.LAH81|04 |A401|5766023019885482064|2107.ACA91|DC |DC01|5766023019768041790|     0|   0|
2107.LAH81|04 |A401|5766023019885482064|2107.ACA91|DD |DD01|5766023019768041790|     0|   0|
2107.LAH81|04 |A401|5766023019885482064|2107.ACA91|D4 |D401|5766023019768041790|     0|   0|
2107.LAH81|04 |A401|5766023019885482064|2107.LAH81|01 |A101|5766023019885482064|     0|   0|
2107.LAH81|05 |A501|5766023019885482064|2107.ACA91|90 |FFFF|5766023019768041790|     0|   0|
2107.LAH81|05 |A501|5766023019885482064|2107.BRX71|0B |4B01|5766023019784820353|     1|   0|0330:0233(13)
2107.LAH81|06 |A601|5766023019885482064|2107.ACA91|DC |DC01|5766023019768041790|     2|   0|0201:0210(13),0330:0330(13)
2107.LAH81|06 |A601|5766023019885482064|2107.ACA91|DD |DD01|5766023019768041790|     2|   0|0201:0210(13),0330:0330(13)
2107.LAH81|06 |A601|5766023019885482064|2107.LAH81|0B |AB01|5766023019885482064|     0|   0|
2107.LAH81|08 |A801|5766023019885482064|2107.ACA91|D4 |D401|5766023019768041790|     1|   0|0330:0330(13)
2107.LAH81|08 |A801|5766023019885482064|2107.ACA91|D8 |FFFF|5766023019768041790|     0|   0|
2107.LAH81|09 |A901|5766023019885482064|2107.ACA91|DA |DA01|5766023019768041790|     2|   0|0201:0210(13),0330:0330(13)
2107.LAH81|09 |A901|5766023019885482064|2107.ACA91|D9 |D901|5766023019768041790|     2|   0|0201:0210(13),0330:0330(13)
2107.LAH81|54 |5410|5766023019885482064|2107.FAW81|B4 |B413|5766023019818374803|     1|   0|0330:0332(13)
2107.LAH81|54 |5410|5766023019885482064|2107.ZA181|54 |5411|5766023019768042708|     2|   0|0201:0230(13),0330:0303(13)
2107.LAH81|55 |5510|5766023019885482064|2107.FAW81|B5 |B513|5766023019818374803|     1|   0|0330:0332(13)
2107.LAH81|55 |5510|5766023019885482064|2107.ZA181|55 |5511|5766023019768042708|     2|   0|0201:0230(13),0330:0303(13)
2107.LAH81|56 |5610|5766023019885482064|2107.ACA91|56 |5612|5766023019768041790|     2|   0|0201:0210(13),0330:0330(13)
2107.LAH81|56 |5610|5766023019885482064|2107.BRX71|56 |5614|5766023019784820353|     1|   0|0330:0233(13)
2107.LAH81|57 |5710|5766023019885482064|2107.ACA91|57 |5712|5766023019768041790|     2|   0|0201:0210(13),0330:0330(13)
2107.LAH81|57 |5710|5766023019885482064|2107.BRX71|57 |5714|5766023019784820353|     1|   0|0330:0233(13)
2107.LAH81|84 |FF84|5766023019885482064|2107.LLB71|84 |FF84|5766023019885482983|     2|   2|0201:0310(17),0330:0242(17)
2107.LAH81|85 |FF85|5766023019885482064|2107.LLB71|85 |FF85|5766023019885482983|     2|   2|0201:0310(17),0330:0242(17)
2107.LAH81|9A |FF9A|5766023019885482064|2107.ACA91|9A |FF9A|5766023019768041790|     2|   0|0201:0210(13),0330:0330(13)
2107.LAH81|9C |FF9C|5766023019885482064|2107.ACA91|9A |FF9A|5766023019768041790|     2|   0|0201:0210(13),0330:0330(13)
```

## CSM_GetSessCpSets(sess,cols,hdr,fmt,delim,sort)

Print copy set details for a given session with data extracted from **GET /CSM/web/sessions/{sess}** query to get volume roles, and data from **GET /CSM/web/sessions/{sess}/copysets** to get copy set details:

```
TLUTHER:/MCECEBC/u/tluther/csmapi:> ./rxcsmapi.rexx -u "CSM_GetSessCpSets('ITSO_MM_LAH81_H1','ID,NAME,DEV,ZATT,SE')"
https://9.155.114.38 (Query: 0.27 sec.): Copysets for session 'ITSO_MM_LAH81_H1': 8
H1-VolID                  |H1-Name|H1-Dev|H1-zAtt|H1-SE|H2-VolID                  |H2-Name|H2-Dev|H2-zAtt|H2-SE
--------------------------|-------|------|-------|-----|--------------------------|-------|------|-------|-----
DS8000:2107.LAH81:VOL:0018|SGA018 |A018  |true   |ESE  |DS8000:2107.ACA91:VOL:D818|SGA018 |D818  |true   |NO
DS8000:2107.LAH81:VOL:0019|SGA019 |A019  |true   |ESE  |DS8000:2107.ACA91:VOL:D819|SGA019 |D819  |true   |NO
DS8000:2107.LAH81:VOL:001A|SGA01A |A01A  |true   |ESE  |DS8000:2107.ACA91:VOL:D81A|SGA01A |D81A  |true   |NO
DS8000:2107.LAH81:VOL:001B|SGA01B |A01B  |true   |ESE  |DS8000:2107.ACA91:VOL:D81B|SGA01B |D81B  |true   |NO
DS8000:2107.LAH81:VOL:001C|SGA01C |A01C  |true   |ESE  |DS8000:2107.ACA91:VOL:D81C|SGA01C |D81C  |true   |NO
DS8000:2107.LAH81:VOL:001D|SGA01D |A01D  |true   |ESE  |DS8000:2107.ACA91:VOL:D81D|SGA01D |D81D  |true   |NO
DS8000:2107.LAH81:VOL:001E|SGA01E |A01E  |true   |ESE  |DS8000:2107.ACA91:VOL:D81E|SGA01E |D81E  |true   |NO
DS8000:2107.LAH81:VOL:001F|SGA01F |A01F  |true   |ESE  |DS8000:2107.ACA91:VOL:D81F|SGA01F |D81F  |true   |NO
```

## CSM_GetSessBackups(sess,hdr,fmt,delim,sort)

Print backup details for a given Safeguarded Copy Session from **GET /CSM/web/sessions/{sess}** query and convert timestamps to local system time:

```
TLUTHER:/MCECEBC/u/tluther/csmapi:> ./rxcsmapi.rexx -u "CSM_GetSessBackups('ITSO_SGC_LAH81')"
https://9.155.114.38 (Query: 0.29 sec.): Available backups for session 'ITSO_SGC_LAH81': 13 backups
RolePair|BackupID  |#CpSets|Valid|BlkExp|Timestamp        |Retention|BackupTime
--------|----------|-------|-----|------|--------D1-------|---------|-----------------------
H1-B1   |1641917107|      8|true |false |22/01/11 17:05:07|unknown  |2022-01-11 17:05:07 CET
H1-B1   |1641913507|      8|true |false |22/01/11 16:05:07|unknown  |2022-01-11 16:05:07 CET
H1-B1   |1641909907|      8|true |false |22/01/11 15:05:07|unknown  |2022-01-11 15:05:07 CET
H1-B1   |1641906307|      8|true |false |22/01/11 14:05:07|unknown  |2022-01-11 14:05:07 CET
H1-B1   |1641902707|      8|true |false |22/01/11 13:05:07|unknown  |2022-01-11 13:05:07 CET
H1-B1   |1641899107|      8|true |false |22/01/11 12:05:07|unknown  |2022-01-11 12:05:07 CET
H1-B1   |1641895507|      8|true |false |22/01/11 11:05:07|unknown  |2022-01-11 11:05:07 CET
H1-B1   |1641891907|      8|true |false |22/01/11 10:05:07|unknown  |2022-01-11 10:05:07 CET
H1-B1   |1641888307|      8|true |false |22/01/11 09:05:07|unknown  |2022-01-11 09:05:07 CET
H1-B1   |1641884707|      8|true |false |22/01/11 08:05:07|unknown  |2022-01-11 08:05:07 CET
H1-B1   |1641881107|      8|true |false |22/01/11 07:05:07|unknown  |2022-01-11 07:05:07 CET
H1-B1   |1641877507|      8|true |false |22/01/11 06:05:07|unknown  |2022-01-11 06:05:07 CET
H1-B1   |1641873907|      8|true |false |22/01/11 05:05:07|unknown  |2022-01-11 05:05:07 CET
```

## CSM_GetSessCmd(sess,hdr,fmt,delim,sort)

Print available action commands for a given session from **GET /CSM/web/sessions/{sess}/availablecommands** query:

```
TLUTHER:/MCECEBC/u/tluther/csmapi:> ./rxcsmapi.rexx -u "CSM_GetSessCmd('DS-MM')"
https://9.155.114.38 (Query: 0.26 sec.): Available commands for session 'DS-MM': 3 commands
Command
---------A1----------
```

```
Enable Copy to Site 1
Start H1->H2
StartGC H1->H2
```

## CSM_RunSessCmd(sess,cmd,parm)

Issue an action command to a given CSM session via **POST /CSM/web/sessions/{sess}** request with cmd in request body parameter:

```
TLUTHER:/MCECEBC/u/tluther/csmapi:> ./rxcsmapi.rexx -u "CSM_RunSessCmd('DS-MM','StartGC H1->H2')"
https://9.155.114.38: Issuing command 'StartGC H1->H2' to session 'DS-MM' ...
Response (1.00 sec.): IWNR1026I [Jan 11, 2022 6:09:57 PM] The StartGC H1->H2 command in the DS-MM session
completed.
```

## CSM_RunHaCmd(cmd,server:port,user,pwd)

Issue a command to manage the CSM HA server relationship via **PUT /CSM/web/system/ha/{cmdtype}/{server}/{username|port}/{password}/{port}**

```
TLUTHER:/MCECEBC/u/tluther/csmapi:> ./rxcsmapi.rexx -u "CSM_RunHaCmd('RECONNECT')"
https://9.155.114.38: Issuing HA server command 'RECONNECT' ...
Response (0.19 sec.): IWNR3050I [Jan 11, 2022 6:15:11 PM] There are no high availability servers configured to
reconnect for the server WINDOWS-PJ1KTM6.
```

## CSM_RunTaskCmd(taskid,cmd,datetime,sync)

Issue a command to manage a task via **PUT /CSM/web/sessions/scheduledtasks/{cmdtype}/{taskid}/{datetime|runat}/{datetime}/{synchronous}** request.

```
TLUTHER:/MCECEBC/u/tluther/csmapi:> ./rxcsmapi.rexx -u "CSM_RunTaskCmd(6,'RUN','2022-01-12T13-00')"
https://9.155.114.38: Issuing task command 'RUN' to task ID '6' ...
Response (0.25 sec.): IWNR2207I [Jan 11, 2022 6:43:14 PM] The scheduled task MMGMGC-Practice is scheduled to
run at 2022-01-12 13:00:00 CET.
```

## CSM_ShowTask(task,hdr,fmt,delim)

Show details for a given scheduled task ID or name from **GET /CSM/web/sessions/scheduledtasks** query and list action steps with details and 9 most recent task messages.

```
TLUTHER:/MCECEBC/u/tluther/csmapi:> ./rxcsmapi.rexx -u "CSM_ShowTask('ITSO_SGC_H3_LAH81')"
https://9.155.114.38 (Query: 1.26 sec.): CSM scheduled task details for 'ITSO_SGC_H3_LAH81'
Entry           |Value
----------------|------------------------------------------------------------
Name            |ITSO_SGC_H3_LAH81
ID              |27
Description     |
Schedule        |Every 0D 01H 00M
RunTaskOnSuccess|0
RunTaskOnFailure|0
PePackageOnError|false
Actions         |4
 *Action(1)     |sessionName=ITSO_MGM_ACA91,command=SuspendH2H3
 *Action(2)     |sessionName=ITSO_MGM_ACA91,state=SuspendedH2H3,timeout=5 min
 *Action(3)     |sessionName=ITSO_SGC_H3_LAH81,command=Backup
 *Action(4)     |sessionName=ITSO_MGM_ACA91,command=Start H1->H2->H3
State           |Disabled
NextRun         |
LastRun         |22/01/05 12:21:31
Messages        |5410
 *Msg(5410)     |22/01/05 12:21:51:IWNR2212I:
 *Msg(5409)     |22/01/05 12:21:51:IWNR1026I:
 *Msg(5408)     |22/01/05 12:21:41:IWNR1026I:
 *Msg(5407)     |22/01/05 12:21:34:IWNR2220I:
 *Msg(5406)     |22/01/05 12:21:34:IWNR1026I:
 *Msg(5405)     |22/01/05 12:21:31:IWNR2211I:
 *Msg(5404)     |22/01/05 11:21:49:IWNR2212I:
```

```
 *Msg(5403)      |22/01/05 11:21:49:IWNR1026I:
 *Msg(5402)      |22/01/05 11:21:44:IWNR1026I:
```

## Rexx Framework function overview

Following sections provide a brief overview of the available functions in the Rexx framework, which can be modified and reused as required.

### Core functions

- InitializeVars:
  Initialize global vars used throughout the program. Default definitions can be hard coded here.
- ParseArgs:
  This will parse the execution parameters and update initialized settings as required
- Usage:
  Print the execution parameter usage description and example
- Main
  Execute the request or function if parameters are specified, or run the hard coded main functions. Those should be customized to whatever should be done by default when no parameters are specified.

### CSM functions

The CSM functions are examples to demonstrate how you can utilize the wrapper and helper functions of the framework to easily perform HTTP requests and process the CSM server responses.

- **CSM_SessOverview**(hdr,fmt,delim,sort):
  Query & print formatted output of a (short) CSM session overview
- **CSM_SysOverview**(hdr,fmt,delim,sort):
  Query & print formatted output of a CSM Storage System overview
- **CSM_PathOverview**(hdr,fmt,delim,sort):
  Query & print formatted output of PPRC paths between Storage Systems
- **CSM_TaskOverview**(hdr,fmt,delim,sort):
  Query & print formatted output of a CSM scheduled task overview
- **CSM_GetSysPaths(sys**,hdr,fmt,delim,sort):
  Query & print formatted output of PPRC paths for a given storage system
- **CSM_GetSessCpSets(sess**,cols,hdr,fmt,delim,sort):
  Query & print formatted output of copy sets for a given CSM session. Cols is an optional comma separated list of columns to be displayed for each volume. Valid columns are: *ID*, *NAME*, *DEV*, *SEfficient*, *ZATTached*, *WWNN*, *PROTected*. The volume columns are displayed in specified order, default is *ID*, *NAME*, *DEV*. Invalid column names will be skipped.
  Note: DEV will show the z/OS device address as defined on the zOS system connected to the CSM server. Without CSM zOS Host connection or necessary zOS APAR, DEV may show N/A.
- **CSM_GetSessBackups(sess**,hdr,fmt,delim,sort):
  Query & print formatted output of available backups for a given CSM session (Safeguarded Copy session types)
- **CSM_GetSessCmd(sess**,hdr,fmt,delim,sort):
  Query & print formatted output of available commands for a given CSM session
- **CSM_RunSessCmd**(**sess**,**cmd**,parm):
  Wrapper to issue an action command to a CSM session. The return code of the function depends on the response message code type. An optional command parameter can be defined (e.g. the backup ID if using commands working with specific backups), and blanks are properly converted to %20 automatically. Valid session commands depend on the session state and status. Available session commands can be queried with CSM_GetSessCmd.

- **CSM_RunHaCmd**(**cmd**,remoteserver:port,user,pwd):
  Wrapper to issue an action to the CSM HA server configuration. A valid action command is required, and server plus optional port may be required for the action. Valid action commands are: *SetAsStandby*, *AddStandby*, *Reconnect*, *Takeover*, *RemoveStandby*. User and password are only required for *AddStandby* command, which will configure the remote server as new Standby server for the local active server. *SetAsStandby* will configure the local active CSM server as new Standby of the specified remote server. The port is optional and only required if different from default port. It must be separated with : from the server name or IP address.
  **Attention**: The specified user and password are not masked and may be printed to logs depending on the framework usage settings. Use the *SetAsStandby* command action preferably to define a HA relationship, which does not require username or password.
- **CSM_RunTaskCmd**(**taskid**,**cmd**,datetime,sync):
  Wrapper to Issue an action command to a CSM scheduled task ID. Supported commands are *RUN*, *ENABLE*, *DISABLE*, *DELETE*. The optional Datetime value must be in the form 'yyyy-mm-ddThh-mm. If the *RUN* command should be issued synchronously, specify sync = 1 or 'ON'. Default is 0 or 'OFF'.
- **CSM_ShowTask**(**task**,hdr,fmt,delim):
  Query & print formatted output of task details for a given task name or ID. It will list detailed task actions and up to 9 most recent task messages if available.

## Wrapper functions to utilize the framework in a simplified way

The wrapper functions are developed to simplify utilization of the web enablement toolkit functions to issue http requests and process JSON responses as provided by the CSM server.

- **HTTP_sendRequest(reqType,reqPath,**reqBody**,**outfile,a2eResBody**)**:
  Wrapper function to handle connection setup & test (using global settings), as well as provided request setup and submit. Request type can either be *GET*, *PUT*, *POST*, *DELETE*, *HEAD*. Request path is the URI to the requested service. Request body is a string containing additional data that may be necessary for the request. Request output is saved to outfile if specified. The response translation a2eResBody setting can optionally be enforced. Otherwise the A2E translation is per default enabled (1 or 'ON') and if "/download" pattern is found in request path, the A2E translation will be disabled (0 or 'OFF') since binary stream data is expected from CSM server.
- **JSON_print(jsontext,**filter**)**:
  Wrapper function to init the JSON parser and format the provided JSON text in a stem variable **g.resData**. Input is the JSON text as received from server and an optional filter can be used to limit the formatted output to specific named entries only. The filter setting will be used only for root entries and each entry must be quoted to allow explicit comparison. Filter example: '"name","description","state"'.
- **JSON_parse(jsonTextBody):**
  Wrapper function to init the JSON parser and parse the input text body, which should be syntactically correct JSON text.

## HTTP functions

These functions are used by the wrapper functions as required. The HTTP functions are customized for the specifics required by CSM server Rest API.

- HTTP_getToolkitConstants:
  Init constants used by the web toolkit
- HTTP_requestToken:
  Wrapper function to request a token for subsequent requests from server
- HTTP_setToken:
  Wrapper function to set or update Token in Header for request
- HTTP_init:
  Create a handle of the designated type (Connection or Request)
- HTTP_setupConnection:
  Sets the necessary connection options.
- HTTP_connect:
  Connect to the configured domain to obtain the connection handle
- HTTP_disconnect:
  Disconnect from configured domain
- HTTP_terminate:
  Release the designated Connection or Request handle
- HTTP_setupRequest:
  Sets the necessary request options
- HTTP_request:
  Make the configured Http request
- HTTP_surfaceDiag:
  Surface input error information
- HTTP_setRequestAuth:
  Set Basic Authentication for request. Note: For CSM API this is only needed if the authentication token process is not used for server requests. This is not used currently by the framework.
- HTTP_setRequestHeaders:
  Add appropriate headers to request
- HTTP_setRequestBody:
  Set necessary attributes for the request body
- HTTP_setRespHdrBody:
  Set options and variables for the response Header and body
- HTTP_TranslateBody:
  Enables or disables the translation of the response body from ASCII to EBCEDIC. This is required if text data is returned that should be parsed later on. This must be disabled if binary stream data is expected, otherwise the byte stream becomes unusable.
- HTTP_isError:
  Check the input processing codes.

## JSON parser functions

These functions are used by the wrapper functions or by the CSM_ functions as required to process the server response.

- JSON_initParser:
  Initializes the global parser handle.
- JSON_searchAndDeserializeData:
  Search for specific values and objects in the parsed response body and deserialize them into the g.resData. stem variable

- JSON_findValue:
  Searches the appropriate portion of the parsed JSON data (that is designated by the objectToSearch argument) for an entry whose name matches the designated searchName argument. Returns a value or handle, depending on the expectedType.
- JSON_getType:
  Determine the Json type of the designated search result.
- JSON_getTypeName:
  Helper function to determine the Json type name from the provided constant value.
- JSON_getNumElem:
  Determine the number of elements contained in the input handle
- JSON_getObjEntry:
  Return a handle to the designated entry of the object designated by the input handle.
- JSON_getArrEntry:
  Return a handle to the designated entry of the array designated by the input handle.
- JSON_getValEntry:
  Return value of a string or number handle
- JSON_getBoolEntry:
  Return the boolean value of a handle
- JSON_termParser:
  Cleans up parser resources and invalidates the parser instance handle
- JSON_isNotFound:
  Check the input processing codes
- JSON_isError:
  Check the input processing codes
- JSON_surfaceDiag:
  Surface input error information

## JSON print functions

These functions are used by the wrapper functions as required. They may be used recursively to format and display nested JSON objects.

- printtype:
  Call the specific printing routine for the json data type.
- printobject:
  Handle printing for a json object type
- printarray:
  Handle printing for a json array type
- printnumber:
  Handle printing for a json number type
- printboolean:
  Handle printing for a json boolean type
- printstring:
  Handle printing for a json string type
- dq:
  Quote the argument for printing functions
- print:
  Prefix the input string with the proper indentation and print it

## Helper functions

These functions are used by various other functions required.

- CheckCsmCmdResp:
  Validate JSON response for an issued command to the CSM server. Will return non 0 if message code was not 'I' type: 4 if 'W' message, 8 if 'E' message, 12 if 'S' message, 16 if unknown message identifier, -1 if error while processing the server response
- InitTable:
  Init the global stem used to compose a formatted output table as used in CSM_ functions.
- PrintTable:
  Print a formatted table from the global output stem as used in CSM_ functions.
- updtCredentials:
  Obtain user credentials for CSM Rest API. Get them from a DS or file. If the specified DS or file does not exist, allocate a new one. If no credentials found and Rexx is executed in foreground, prompt user for credentials and save them with encoding. The encoding will be via a private key that shuffles and translates all chars. The private key will be loaded once from the specified private key file. If no private key found, a new one will be created and saved. If a request token was received, it will also be saved in the credential file, so it can be reused on subsequent requests while valid.
- allocateDD:
  Allocate existing DSN or file or create new if not existing. The specified type will be recognized from the name. Allocation options will be used when a new file must be allocated.
- getAlocFile:
  Get allocated DSN or path from provided DD name
- checkUSSFile:
  Verify the state of the given USS path/filename
- getISPFcred:
  Create or use existing ISPF panel member to prompt credentials without echo
- cipher:
  Encode or decode the given string (default is encode). If KEYGEN is specified, a new private key is returned. This function can be changed if different encryption methods should be used by the Rexx framework. In case your Z system has crypto cards installed, you can utilize Rexx calls to generate strong encryption keys and use them for encoding and decoding. See following example on the IBM community:
  https://community.ibm.com/community/user/ibmz-and-linuxone/blogs/eysha-shirrine-powers2/2020/03/25/rexx-sample-aes-clear-key-generate-write-to-ckds-encrypt-and-decrypt
- ConvBlanks:
  Converts blanks to %20 and vice versa for parameters to be used in request body or URLs.
- ConvUnixTime:
  Converts Unix Epoche timestamp to YYYY/MM/DD HH:MM:SS. Unix time is the number of seconds since midnight 1-1-1970. CSM timestamps are Unix UTC timestamps in milliseconds. Use this function to convert CSM server timestamps to a readable format in local system time.
- TimeOffset:
  Get the local time offset from GMT in mm:ss
- fatalError:
  Surfaces the input message, and returns a failure code

- cleanup:
  Exit function. Cleanup a connection and possibly a request handle if active. Then free all allocated DDs and post optional error message prior exit.