Co:Z® Co-Processing Toolkit for z/OS

Co:Z Dataset Pipes - User's Guide

V 4.5.1 Edition

Published October, 2017

Table of Contents

1. Introduction

Co:Z Dataset Pipes are utilities that access z/OS data and services.

There are commands for accessing z/OS data sets, Unix Systems Services files, the JES spool, and more.

Co:Z Dataset Pipes can be used in the following modes:

- **z/OS Unix System Services Integration:** A z/OS unix process accesses local z/OS services

  The Dataset Pipes commands can be invoked from Unix System Services directly (interactively, or from a shell script) or used as shell commands within a *Co:Z Batch* job step.

- **z/OS Hybrid Batch:** A z/OS jobstep launches a remote process on a target system

  The Co:Z Launcher starts a shell process on a distributed system, redirecting its input and output to traditional z/OS datasets or spool files.

  The Dataset Pipes client commands can be used by the remote process to reach back into the launching jobstep to access z/OS files, datasets and other services.

  The target may be another z/OS system with Co:Z installed.

- **z/OS Remote Services:** A remote client initiates a connection to z/OS

  A Unix, Windows or remote z/OS system can use the Dataset Pipes client commands to initiate an SSH connection to a z/OS server.

  Commands can be run individually (each with its own SSH connection), or through a durable connection initiated by the remote system with the `cozcontrol` command. When using `cozcontrol`, a single dataset pipes connection is used for multiple command invocations.

Features:

- Pipe input to an MVS dataset or a POSIX file (**todsn** and **tofile**)
- Pipe output from an MVS dataset or a POSIX file (**fromdsn** and **fromfile**)
- Remote execution over an SSH connection
- Supports any z/OS dataset which can be opened in sequential, record mode by the `fopen()` C-library routine. This includes:
  - MVS sequential datasets (QSAM, BSAM)
  - PDS and PDSE members
  - VSAM files (processed in sequential mode)
  - SYSOUT datasets, including the MVS internal reader
- Supports text or binary conversion via flexible line-termination rules:

  - Cr, Lf/Newline, CrLf, Cr and/or Lf, RDW, none, user-defined-string
- Supports flexible record padding / overflow rules:
  - wrap, flow, truncate, error
- Codepage translation via high-performance z/OS conversion services
- Can specify additional `fopen()` options and dynamic allocation keywords
  - keywords supported by **BPXWDYN** can be used to customize dataset allocation
  - allows for SYSOUT, writers or MVS internal reader
- User and/or system profile can be used to automatically supply conversion options based dataset name matching.

2. Dataset Pipes Installation and Configuration

2.1 Dataset Pipes Installation by Use Case

The installation steps vary depending on the usage of the Dataset Pipes commands.

- **z/OS Unix System Services Integration:**

  Follow the instructions for *Co:Z Toolkit for z/OS*. Because the Dataset Pipes commands are only running on z/OS, no additional configuration is needed. There is no specific Dataset Pipes configuration.

- **z/OS Hybrid Batch:**

  When Dataset Pipes commands are run with the Co:Z Launcher in a z/OS Hybrid Batch use case, installation is required for the *Co:Z Toolkit for z/OS* and the *Co:Z Target System Toolkits*. No specific Dataset Pipes configuration is required, but see the *Co:Z Launcher User's Guide* for additional configuration options.

- **z/OS Remote Services:**

  In addition to the *Co:Z Toolkit for z/OS* and *Co:Z Target System Toolkits*, the dspipes subsystem and dspipes session config options must also be configured. See the following sections for more detail.

2.2 z/OS Remote Services Quick Start

After completing the installation of the *Co:Z Toolkit for z/OS* and the *Co:Z Target System Toolkit* on the remote system, the following are the minimum steps to get started using z/OS Remote Services. Replace <COZ_INST> with the installation directory of the toolkit. For more detailed information, see the remaining sections in this chapter.

On z/OS:

1. Edit `/etc/ssh/sshd_config` and add the following line along with other subsystem configuration.

```
Subsystem dspipes <COZ_INST>/bin/dspipes.sh
```

   Restart SSHD.

```
kill -HUP `cat /var/run/sshd.pid`
```

2. Copy the sample configuration files to `/etc/ssh`:

```
cp <COZ_INST>/samples/dspipes.site.rc /etc/ssh/dspipes.rc
chmod 755 /etc/ssh/dspipes.rc
cp <COZ_INST>/samples/dspipes_site_config /etc/ssh/dspipes_config
```

```
chmod 0644 /etc/ssh/dspipes_config
```

3. Edit `/etc/ssh/dspipes_config`. Uncomment and set `server-host` to an externally recognized IP address or hostname for the z/OS server. Uncomment and set `server-ports`, if the default is not applicable.

On the remote system:

Assuming the Co:Z Target System Toolkit has been installed on a unix server with the PATH configured to the Dataset Pipes command executables:

1. Verify that the remote system can connect to the z/OS server using SSH.

```
ssh user@host uname -a
```

2. Interactively test a Dataset Pipes command with a single SSH connection. This command lists the files in the user's z/OS home directory.

```
cozclient -ssh user@host ls -alt
```

3. Interactively start a durable connection using **cozcontrol**, execute a few Dataset Pipes commands on z/OS, and end the session.

```
cozcontrol start -ssh user@host
cozclient ls -alt
fromdsn 'SYS1.MACLIB(ACB)' > /tmp/test.txt
todsn 'user.test.dataset' < /tmp/test.txt
cozcontrol stop
```

For more examples, see *Appendix A, Command Reference - Dataset Pipes* and *Chapter 4, z/OS Remote Services Examples*.

2.3 z/OS Remote Services Configuration

This section provides additional detail about the steps summarized in the quick start section above.

Configuring the Dataset Pipes subsystem

To run Dataset Pipes commands initiated by a remote client, a subsystem must be configured in your z/OS OpenSSH server.[1]

This is done by updating the `sshd_config` file, typically located at `/etc/ssh/sshd_config`.[2]

[1]SSH user subsystems are, like all SSH remote commands, executed in a process under the authenticated client userid, so normal z/OS user security determines what resources can be accessed.

Find the line "Subsystem" which defines the `sftp` subsystem. Immediately following the sftp line add this:

```
Subsystem dspipes <COZ_INST>/bin/dspipes.sh
```

(where <COZ_INST> is the directory where Co:Z Toolkit is installed).

If OpenSSH **sshd** was running prior to editing `sshd_config`, it should be reinitialized. This can be done by sending `SIGHUP` to the running process. The pid for this process is typically in the file `/var/run/sshd.pid`:

```
kill -HUP `cat /var/run/sshd.pid`
```

Dataset Pipes Configuration overview

The following table describes how a Co:Z Dataset Pipes Server session is started and outlines the sequence of configuration steps that occur prior to the establishment of the session. Details on these configuration steps follow the table.

Table 2.1. Dspipes Server initialization steps

Step	Configuration	Notes
1	$COZ_HOME/bin/dspipes.sh	This shell script is executed by z/OS OpenSSH **sshd** upon a request for an Dataset Pipes server subsystem. *This file should not be modified by the installation*, but you may want to review the comments at the beginning of the script. This script will execute the site-wide and user-specific rc scripts and configuration files (see following steps).
2	/etc/ssh/dspipes.rc	Site-wide environment variable configuration.
3	$HOME/.ssh/dspipes.rc	User specific environment variable configuration. Can contain customized log file location, logging and tracing options, etc. The location of this file may be changed by setting the $DSPIPES_USER_RC environment variable.
4	$HOME/.ssh/dspipes_config	User-specific configuration settings. User customized options may be specified here. The location of this file may be changed by setting the $DSPIPES_USER_CONFIG environment variable.
5	/etc/ssh/dspipes_config	Site-wide configuration settings.

[2]It is sometimes convenient to set up a *test* OpenSSH server where this subsystem can be easily added. Instructions for doing this can be found in the Co:Z Installation and Release Notes.

Sitewide Dataset Pipes configuration

The Dataset Pipes server can be configured with system-wide defaults by creating and configuring the file `/etc/ssh/dspipes.rc`. A sample file (`dspipes.site.rc`) is provided in `<COZ_INST>/samples`, and should be copied to the `/etc/ssh` directory:

```
cp <COZ_INST>/samples/dspipes.site.rc /etc/ssh/dspipes.rc
chmod 755 /etc/ssh/dspipes.rc
```

Sample site dspipes.rc file

```
#!/bin/sh
# Set site-wide environment variables for dspipes server.
# Place this sample as an executable script in file: /etc/ssh/dspipes.rc

# The following are the default locations for user level configuration files.
#DSPIPES_USER_RC=$HOME/.ssh/dspipes.rc ❶
#DSPIPES_USER_CONFIG=$HOME/.ssh/dspipes_config ❷

# The following defines a directory name (without trailing slash) where
# log files will be created, rather than /tmp or $TMPDIR. Setting this
# variable and regular cleanup of this directory are recommended.
#export DSPIPES_LOGDIR=
```

❶ In some cases, users may not have access to individual `$HOME` directories or it may be desirable to have all user configuration files centralized. In this case, the environment variable `DSPIPES_USER_RC` can be specified to provide an alternate file name for the user .rc file in a common, readable location. For example, to specify a common directory for all user configuration files, set the following: `DSPIPES_USER_RC=/usr/share/coz/$LOWER_LOGNAME.dspipes.rc`

To disable the usage of user specific dspipes.rc files for all users, `DSPIPES_USER_RC` can be set to a dummy file name (e.g: `/dummy`).

Note that the z/OS Unix System Services `$LOGNAME` environment variable holding the current username is in uppercase. As this is not always consistent with other POSIX style usage, the **dspipes.sh** script exports an enviroment variable named `$LOWER_LOGNAME` that downcases the value in `$LOGNAME`.

❷ Additionally, individual user server config files can be similarly located. To learn more about config files, refer to section: *Section 2.4, "z/OS Remote Services Session Config Files"*. By default, user server config files are located at `$HOME/.ssh/dspipes_config`.

Note: The `/etc/ssh/dspipes.rc`, if present, must be marked executable, as must the individual user files.

User specific Dataset Pipes customization

Individual users can override system wide Dataset Pipes properties by creating a profile script, `dspipes.rc`, in their home .ssh directory:

```
# if the user's .ssh does not exist:
mkdir $HOME/.ssh
```

```
chmod 700 $HOME/.ssh

cp <COZ_INST>/samples/dspipes.user.rc $HOME/.ssh/dspipes.rc
chmod u+x $HOME/.ssh/dspipes.rc
```

Sample user dspipes.rc file

```
#!/bin/sh
# Place this sample as an executable script in file: $HOME/.ssh/dspipes.rc

# Product support personnel may request that you uncomment one or more of
# the following variables to enable tracing. These options should be set
# in user specific dspipes.rc files rather than globally in the site-wide
# dspipes.rc

# Setting COZ_LOG enables tracing for CozServer session level tracing.
# The default is N, Notice.
#export COZ_LOG=T ❶

# Setting COZ_LOG_CMD enables tracing for dspipes commands running on the
# server (fromdsn, cozclient, etc). The default is N, Notice. Command
# tracing can alternately be enabled with the -L option on most dataset
# pipes commands.
#export COZ_LOG_CMD=F ❷

# Setting COZ_LOG_CMD_DUP to true (default is false), duplicates tracing enabled by
# COZ_LOG_CMD to the session log. This is recommended when requesting support from
# Co:Z support personnel because all logging for a problem is captured in a single
# file.
#export COZ_LOG_CMD_DUP=true ❸
```

❶ When logging is enabled by setting this variable, log files are created for every Dataset Pipes server session. Each session is an SSH session. See *the section called "Logging Configuration for Dataset Pipes"* for additional information.

❷ When command logging is enabled by setting this variable, Co:Z logging messages are redirected back to the client for all Dataset Pipes commands executed by the client. The optional `-L` switch can be specified on Dataset Pipes commands to enable logging for a single command.

❸ When set to `true`, command logging is redirected back to the client and duplicated to the session log.

Logging Configuration for Dataset Pipes

Log files are created for every Dataset Pipes server session. When specifying `-ssh` on an individual Dataset Pipes command such as **fromdsn**, the log file contains logging only for the single command. When using **cozcontrol** to define a durable session used by multiple Dataset Pipes commands, the log file contains logging for all commands using the durable session. If the log file remains empty due to the logging level set, the file is deleted when the session ends. The log file is of particular interest when a problem is encountered and additional error detail is needed. The following sections define how to control the logging destination as well as logging levels.

Logging Destination

By default, log files are written to the `/tmp` directory (or the directory specified by the `TMPDIR` environment variable, if it is set). To change this directory default for all users, modify `/etc/ssh/dspipes.rc` as needed. Exporting `DSPIPES_LOGDIR` changes the directory log files are written to. The session log file name is generated with the following pattern: `dspipes.<userid>.<...>.log`.

The directory containing the log files must be cleaned up and monitored for space; however, it is important to keep these files for some period of time in order to allow support personnel to review the session log file for diagnostic information when investigating a problem.

Logging Level

The logging level is controlled by exporting the following environment variables: `COZ_LOG`, `COZ_LOG_CMD`, and `COZ_LOG_CMD_DUP`. In order to diagnose a problem for an individual user, create a `dspipes.rc` file in their individual `.ssh` directory, setting these variables as directed by product support personnel.

- `COZ_LOG`

  When set, enables Dataset Pipes server level logging, excluding logging of the server side execution of Dataset Pipes commands (fromdsn, todsn, cozclient, etc.).

  When the `COZ_LOG` environment variable is not specifically set, the default is `N` which logs error, warning and notice messages. The Co:Z support team may direct setting this variable to one of E, W, N, I, D, T or F for Error, Warning, Notice, Informational, Debug, Trace, or Fine level tracing.

- `COZ_LOG_CMD`

  When set, enables logging of the server side execution of Dataset Pipes commands (fromdsn, todsn, cozclient, etc.). Logging for Dataset Pipes commands is redirected back to the client as stderr. Alternatively, logging can be configured for a specific Dataset Pipes command by specifying the `-L` option on the command. See *Appendix A, Command Reference - Dataset Pipes* for additional information.

  When the `COZ_LOG_CMD` environment variable is not specifically set, the default is `N` which logs error, warning and notice messages. The Co:Z support team may direct setting this variable to one of E, W, N, I, D, T or F for Error, Warning, Notice, Informational, Debug, Trace, or Fine level tracing.

- `COZ_LOG_CMD_DUP`

  The default for this variable is `false`. Setting this variable to `true` duplicates Dataset Pipes command stderr to the client and the server session log. Setting this option to `true` in a user's `dspipes.rc` file captures the most complete set of information in one server session log file.

2.4 z/OS Remote Services Session Config Files

The file `/etc/ssh/dspipes_config` can be used to customize the options available for a **cozcontrol** durable session. The permissions for this file should be `0644`. While user's can override these properties, they are intended to be site-wide properties.

A sample file (`dspipes_site_config`) is provided in `<COZ_INST>/samples`, and should be copied to the `/etc/ssh` directory:

```
cp <COZ_INST>/samples/dspipes_site_config /etc/ssh/dspipes_config
chmod 644 /etc/ssh/dspipes_config
```

Table 2.2. Durable session Dataset Pipes server properties

Property	Description
server-ports	Default: 8050-8058. Identifies the ports the server can use to set up its socket listener. This property is not used when tunneling is configured on **cozcontrol start**.
server-host	Default: gethostname(). The external address of the Dataset Pipes server running on z/OS. Used by the **cozcontrol** and client Dataset pipes commands to send messages to the socket listener established when the Dataset Pipes server is started. This option is not used for tunneled connections.
inactivity-interval-minutes	Default: 15 minutes. The Dataset Pipes server will shutdown its socket server and end the SSH connection when the inactivity-interval-minutes expires due to no Dataset Pipes command activity during the durable session.

3. General Dataset Pipes Examples

This chapter contains common examples for using Dataset Pipes. The commands in this section can be used in three different environments (use cases):

- **z/OS Unix System Services Integration:** from any z/OS Unix shell (see *Section E.2, "Using the z/OS Unix Shell"*) including Co:Z Batch,
- **z/OS Hybrid Batch:** from a shell process executing on a distributed system by the Co:Z Launcher, or
- **z/OS Remote Services:** from a remote client initiatinig a connection to z/OS using a **cozcontrol** durable SSH connection. Optionally, these commands can be modified to specify the target system connection information directly on the command by adding the `-ssh` option. See *Chapter 4, z/OS Remote Services Examples*.

These examples assume that you have installed and configured the *Co:Z Toolkit for z/OS* on your z/OS server, the *Co:Z Target System Toolkits* on non-z/OS servers, and in the z/OS remote services use case, completed *the section called "Configuring the Dataset Pipes subsystem"* on the target z/OS server.

For questions or to suggest new examples for this chapter, please visit the *Dovetailed Technologies z/OS Forum*

3.1 Copy an MVS dataset to a zFS file

```
fromdsn //MVS1.INPUT.DATASET > /home/user/mydata
```

The **fromdsn** command reads an MVS dataset and converts it to a stream of bytes written to `stdout`. The above command redirects ('>') this output to a zFS file. With the default options for **fromdsn**:

- Trailing pad characters (default is spaces) will be removed from the dataset records
- Linefeeds (EBCDIC "newline") characters will be added to the end of each record

3.2 Copy a zFS file to an MVS dataset

```
todsn //MVS1.OUTPUT.DATASET < /home/user/myfile
```

The zFS file is redirected to `stdin` of the **todsn** command which converts the data to records written to the MVS dataset. The default options for **todsn** are in effect:

- Input lines will be broken on CR, LF, or CRLF.
- If the dataset is new, then its default attributes will be `"recfm=VB,lrecl=1028"`.
- Lines longer than allowed by the dataset will be wrapped onto multiple records.

3.3 Copy an MVS dataset (PDS member) to a zFS file

```
fromdsn '//mvs1.my.lib(member1)' > /home/user/member1
```

The **fromdsn** command reads an MVS dataset and converts it to a stream of bytes written to `stdout`. The above command redirects ('>') this output to a zFS file. With the default options for fromdsn:

- Trailing pad characters (default is spaces) will be removed from the dataset records
- Linefeeds (EBCDIC "newline") characters will be added to the end of each record
- The single quotes are required to prevent the Unix shell from interpreting the parentheses as meta characters.

3.4 Copy a zFS file to a PDS member

```
todsn '//MVS1.MYLIB.DATA(MEMBER1)' < /home/user/myfile
```

The single quotes are required so that the parentheses will not be interpreted as shell meta-characters.

3.5 Specifying dataset names

```
todsn //userid.test.data < /home/user/myfile
todsn -r //test.data < /home/user/myfile
```

- By default, dataset names are assumed to be fully-qualified.
- The `-r` option can be used to automatically add a prefix of the current userid. Assuming that the current userid is "userid", the to above commands use the same dataset.
- Dataset names are always upper case, but upper or lower case names may be given.
- Dataset names that include PDS member names should be enclosed in single quotes, so that the parentheses will not be interpreted as shell meta characters. Quoting the dataset name does not imply anything more; the `-r` option may still be used to indicate that the userid should be added as a prefix.

3.6 Copy an ASCII zFS file to an EBCDIC MVS dataset

```
todsn -s iso8859-1 -r //my.dataset < /home/user/ascii.txt
```

- The `-s` option names the source codepage(charset) used to convert the data.

- The `-t` option may be used to specify the target codepage.
- If either `-s` or `-t` is omitted, they default to the current codepage for the process's locale, which is commonly "IBM-1047" (EBCDIC, Latin).
- The arguments to `-s` and `>-t` may also be numeric CCSIDs.
- If the same effective CCSID is specified as both the source and target, then no translation is performed.
- The IBM z/OS Unicode Translation service (see *Section E.5, "The z/OS Unicode Translation Services"*), is used for all codepage conversions. Starting with z/OS 1.6, this service is configured and enabled by default, but your environment may need to be customized to include specific codepage that you wish to use. If the requested codepage conversions are not available, then Dataset Pipes will try to fall back and use the `iconv()` C-library routine.

3.7 Copy to an MVS dataset, overriding target DCB attributes

```
todsn -o 'recfm=fb,lrecl=80' //MVS1.DATASET1 < /home/user/myfile
```

The `-o` option is used to provide additional options to the `fopen()` API. (see *Section E.3, "The z/OS C library fopen() routine"*), which is used by **todsn** to open the output dataset. The base `fopen()` options used by todsn to open output datasets is `"rb,type=record,noseek"` Since fixed length records are called for in this example, todsn will pad any short records with spaces. (The pad character can be overridden using the `-p` option).

3.8 Copy to an MVS dataset, truncating long lines

```
todsn -w trunc //MVS1.DATASET1 < /home/user/myfile
```

The `-w` option is used specify how to handle lines longer than the maximum record length of the target dataset. The default is to wrap long lines to a new record. Specify `trunc` to cause long lines to be truncated, or `error` to cause the command to fail if a long line is encountered.

3.9 Copy an MVS dataset using DISP=SHR

```
fromdsn -x shr //mvs1.input.dataset > /home/user/mydata
```

The default allocation status used by `fopen()` in "read" mode is DISP=OLD. The `-x` option can be used to specify BPXWDYN allocation keywords (see *Section E.4, "The z/OS BPXWDYN dynamic allocation service"*). In this example, the keyword `shr` is used to specify a allocation status of "share", which allows for multiple jobs to read the same dataset simultaneously.

3.10 Copy one MVS dataset to another

```
fromdsn //mvs1.input.dataset | todsn //mvs1.output.dataset
```

The **fromdsn** reads the input dataset and converts it to a stream of bytes which is piped into the **todsn** command which converts that stream of bytes to the output dataset. If the output dataset is new, then the default attributes of `"recfm=vb,lrecl=1028"`. Existing DCB attributes are used if the output dataset already exists. Default line-termination and wrap rules apply, which fine for text data.

3.11 Copy one MVS dataset to another using the same attributes

```
fromdsn //mvs1.input.dataset |
  todsn -x 'new like(mvs1.input.dataset)' //mvs1.output.dataset
```

The `-x` option is used to specify the "new" and "like" BPXWDYN allocation keyword, which copies attributes (DCB, SPACE, etc) from a model dataset to allocate the new output dataset. Newline characters are, by default, used as record delimiters, so this command is only appropriate for text datasets.

3.12 Copy one MVS non-text dataset to another

```
fromdsn -k -l rdw //mvs1.input.dataset |
  todsn -l rdw -x 'new like(mvs1.input.dataset)' //mvs1.output.dataset
```

The `-l rdw` option is used on both the **fromdsn** and **todsn** commands to indicate that four byte record-descriptor-words (RDW) should be used in the piped stream to indicate record boundaries. The fromdsn `-k` option specifies that pad characters should not be trimmed from the end of records (trimming is the default for fixed-length records).

3.13 Translate a file to the ISO8859-1 codepage from the default z/OS process codepage

```
fromfile -t ISO8859-1 myfile.txt > myfile_win.txt
```

- The `-t` option may be used to specify the target codepage.

3.14 Copy a zFS file to a new location, creating any

missing path components

```
tofile -p /home/user/newdir/myfile < myfile
```

- The `-p` option make the path components to filename if they don't exist (ala mkdir -p).

3.15 Copy user input to the end of an exiting dataset

```
todsn -a //userid.test.data
```

- Since the todsn command gets its input from `stdin`, entering the command without a pipe will cause it to read from the terminal. The user can type input lines, ending it `ctrl-d` which signals an end-of-file.
- The `-a` option changes the base `fopen()` options to `"ab,type=record,noseek"`, which opens the file in append (aka "mod") mode. This option can of course be used with pipes as well.

4. z/OS Remote Services Examples

This chapter contains common examples for using Dataset Pipes, specifically from a remote system. These examples assume that you have installed and configured the *Co:Z Toolkit for z/OS* on your z/OS systems, the *Co:Z Target System Toolkits* on non-z/OS systems, and completed *the section called "Configuring the Dataset Pipes subsystem"* on the target z/OS server.

For a complete set of Dataset Pipes commands that can be executed from a remote system, see *Appendix A, Command Reference - Dataset Pipes* .

4.1 Download a dataset to a unix server over a SSH connection

```
fromdsn -ssh user@zos.myco.com 'hlq.input.dataset' > /tmp/data
```

- Downloads a MVS dataset over a SSH connection.

4.2 Upload a dataset from a unix server over a SSH connection

```
cat /tmp/data | todsn -ssh -p 2222 user@zos.myco.com 'hlq.input.dataset'
```

- Uploads a MVS dataset over a SSH connection.

4.3 Copy a MVS dataset from one z/OS system to another over a SSH connection

```
fromdsn -k -l rdw //mvs1.input.dataset |
  todsn -ssh user@zos2.myco.com -l rdw //mvs2.output.dataset
```

- **fromdsn** is run locally to create a stream of RDW-delimited records that is piped into the **todsn** command.
- The todsn `-ssh` option creates a SSH client connection over which it runs a remote todsn command on the target system.
- The `-ssh` option requires that z/OS OpenSSH is available and configured.

- This example assumes that you have configured SSH authentication keys, since the todsn command does not allow for password prompting.

4.4 Using a durable connection, run simple dspipes commands, end the connection

```
cozcontrol start -ssh user@zos.myco.com
fromdsn 'hlq.dsn' > /home/user/mydata/data1.txt
cat /home/user/mydata/data2.txt | todsn 'hlq.dsn'
cozclient wto "message to console"
cozcontrol stop
```

- The **cozcontrol start** command establishes a SSH connection from unix to the z/OS server.
- The **fromdsn** command downloads a dataset to the unix server
- The **todsn** command uploads a file to a dataset.
- The **cozclient** command write a message to the console.
- Finally, the **cozcontrol stop** command ends the durable SSH connection.

4.5 Using a durable connection, run a pax command on z/OS, download the archive to unix, end the connection

```
cozcontrol start -ssh user@zos.myco.com
cozclient pax -wzvf "//'HLQ.DATA.ARCHIVE'" /home/user/datadir
fromdsn -b 'hlq.data.archive' > /home/user/mydata/data.archive.pax
cozclient tso delete 'data.archive'
cozclient wto "archive complete"
cozcontrol stop
```

- The **cozcontrol start** command establishes a SSH connection to the z/OS server.
- The **cozclient** command executes a **pax** on z/OS to back up a directory to a dataset. Note: pax archives to a dataset only to show a tso delete command in the example.
- The **fromdsn** command copies the dataset backup to the remore system in binary mode.
- The **cozclient** command deletes the dataset backup.
- The **cozclient** command write a message to the console indicating that the archive is complete.

- Finally, the **cozcontrol stop** command ends the durable SSH connection.

4.6 From a remote linux system, start a tunneled durable connection, run dspipes commands, end the connection

```
cozcontrol start -t -ssh user@zos.myco.com
fromdsn 'hlq.dsn' > /home/user/mydata/data1.txt
cat /home/user/mydata/data2.txt | todsn 'hlq.dsn'
cozclient wto "message to console"
cozcontrol stop
```

- The **cozcontrol start** command specifies the `-t` option setting up a SSH control master with local port forwarding.
- Subsequent **todsn**, **fromdsn**, and **cozclient** commands are forwarded over this SSH connection.

4.7 Scripted Co:Z Remote Services example

```
#! /bin/bash

# Sample script: z/OS Remote Services Example
#
# This script expects user@host as a parameter. Using this parameter,
#    1. establishes a cozcontrol control session.
#    2. retrieves a dataset using the socket established on start,
#    3. runs a command on the z/OS server,
#    4. stops the cozcontrol session.
#
# ENVIRONMENT VARIABLE:
#    COZ_LOG
#    COZ_CONTROL_SESSION
#
# SCRIPT VARIABLES:
#    USER_HOST  - initialized with input parameter value
#    SSH_LOG_FILE
#
export COZ_LOG=I

#SSH_LOG_FILE="-E /tmp/ssh.log"

if [ "$1" = "" ]
then
  echo "Usage: user@host required."
  exit
fi
USER_HOST=$1
```

```
# Start a cozcontrol session
./cozcontrol start -ssh $SSH_LOG_FILE $USER_HOST
rc=$?
if [ "$rc" -ne "0" ]; then { echo "cozcontrol start failed. rc=$rc" ; exit 1; } fi

# Retrieve a dataset from the server.
COZ_CONTROL_SESSION=$USER_HOST ./fromdsn 'hlq.dsn' > /tmp/data.txt
rc=$?
if [ "$rc" -ne "0" ]; then { echo "fromdsn failed. rc=$rc" ; exit 1; } fi

./cozclient wto "MESSAGE TO CONSOLE"

# Stop the cozcontrol session
COZ_CONTROL_SESSION=$USER_HOST ./cozcontrol stop
rc=$?
if [ "$rc" -ne "0" ]; then { echo "cozcontrol stop failed. rc=$rc" ; exit 1; } fi
```

- A zero return code from **cozcontrol start** means that the SSH connection to the server has been established. If the ssh connection is not successful, SSH logging can be enabled by adding `-vvv` to the **cozcontrol start** command and reviewing the log captured by setting the script variable `SSH_LOG_FILE`.
- Setting the environment variable `COZ_CONTROL_SESSION` to the user@host parameter on each DatasetPipes command allows multiple copies of the script to be run concurrently by the same user to different target hosts.

4.8 From a Windows workstation, download a MVS dataset over a SSH connection

```
fromdsn -ssh user@zos2.myco.com //mvs1.input.dataset > c:\mydata\data1.txt
```

- **fromdsn.exe** is a Windows program that creates a SSH connection to a remote z/OS host to remotely run the z/OS fromdsn command.
- On Windows, the `-ssh` option requires that the PuTTY **plink** command be installed and available on the PATH.
- fromdsn is also available in source for building on POSIX / Unix systems as part of the Co:Z target server toolkit
- fromdsn.exe has the same arguments and features as the z/OS **fromdsn** command, with the addition of options for specifying the remote z/OS SSH user@host, and optional arguments to SSH / Putty. See the other examples for features of fromdsn that you may remotely use via fromdsn `-ssh`.
- The linemode option `-l` defaults to `crlf` for the Windows client, and the by default the source codepage will be the same as the current Windows codepage.
- The output of the fromdsn command is the converted stream of data, which is redirected ('>') to a PC file.

4.9 From a Windows workstation, upload an MVS dataset (PDS member) over a SSH connection

```
copy c:\upload.txt con: |
  todsn -ssh user@zos.myco.com '//userid.lib.data(mem1)'
```

- The Windows copy command is used to pipe ('|') the contents of a file into the **todsn** command.
- **todsn.exe** is a Windows executable that creates a SSH connection to a remote z/OS host to remotely run the z/OS todsn command.
- On Windows, the todsn `-ssh` options requires that the PuTTY **plink** command be installed and available on the `PATH`.
- todsn.exe has the same arguments and features as the z/OS todsn command, with the addition of options for specifying the remote z/OS SSH user@host, and optional arguments to SSH / PuTTY. See the other recipes in this cookbook for features of todsn that you may use remotely with the Windows SSH client.

Appendix A. Command Reference - Dataset Pipes

- *cozclient(1)*
- *cozcontrol(1)*
- *fromdsn(1)*
- *fromfile(1)*
- *toasa(1)*
- *todsn(1)*
- *tofile(1)*

Name

cozclient — run a zos command from a remote system

Synopsis

```
cozclient [OPTION...] command [command-options...]
cozclient -ssh [ssh-options...] [user]@host [OPTION...] command [command-options...]
cozclient -v
cozclient -h
```

Description

The **cozclient** command allows a remote process to excute the z/OS *command [command-options...]*. Input (`stdin`) to the command is provided by the remote process and Output (`stdout`) from the command is redirected back to the remote process. Error output (`stderr`) from the command can be routed back to the remote client or to the Co:Z Server's stderr stream (if using the Co:Z Launcher).

The z/OS path when executing the command will by default be set to `/bin:$COZ_HOME/bin`.

The **cozclient** command runs in one of the following environments:

- remotely, from a client which was started by Co:Z launcher.
- remotely, from a client that started a durable session to the server using the **cozcontrol** command.
- remotely, from a client-initiated ssh connection: `-ssh` option

Options

`-ssh` [ssh-options...] `[user]@host`
: Specifies a remote invocation of **cozclient** using a client-initiated ssh connection to the given z/OS user@host. The optional stdin, stdout, stderr format options, if specified, must be before the `-ssh` option.

`-h`
: display help and exit.

`-i` stdin_format
: `t`

  stdin sent to the command in text format. Characters are converted from the remote client's codepage to to the active z/OS codepage before being sent to the command.

  `b`

  stdin sent to the command in binary format

  `n`

no stdin is sent to the command. This is the default.

`-o` stdout_format

`t`

stdout from the command is sent to the remote client in text format. Characters are converted from the active z/OS codepage to the remote client's codepage. This is the default.

`b`

stdout from the command is sent to the remote client in binary format

`n`

stdout from the command is discarded

`-e` stderr_format

`t`

stderr from the command is sent to the remote client in text format. Characters are converted from the active z/OS codepage to the remote client's codepage. This is the default when `-ssh` is specified or a **cozcontrol** durable session is used.

`b`

stderr from the command is sent to the remote client in binary format

`s`

stderr from the command is sent to the Co:Z Server's stderr stream (generally SYSOUT). This is the default when using CoZLauncher.

`-v`

display the current version and exit.

Examples

Remote Co:Z Launcher Examples

```
cozclient -in -ot ls -al
```

Run the `ls` command on z/OS. Output is converted to the client codepage and is directed to the remote system's stdout stream.

```
cozclient -in wto "MESSAGE TO CONSOLE"
```

Use the Co:Z toolkit z/OS **wto** command to send a message to the z/OS console.

Remote Client SSH Connection Examples

```
cozclient -ssh user@myzos2.com ls -al
```

Run the `ls` command on z/OS.

```
cat jcl.txt | cozclient -ssh user@myzos2.com submit
```

Submits a job to the internal reader on z/OS. The JCL is contained in the local file `jcl.txt`.

Name

cozcontrol — start/stop a durable connection to the Dataset Pipes subsystem on a target server

Synopsis

```
cozcontrol start [-t]  -ssh [ssh-options...] [user]@host
cozcontrol stop
cozcontrol -v
cozcontrol -h
```

Description

The **cozcontrol** command starts and stops a durable connection to the Dataset Pipes subsystem on a target server.

When starting a durable session, the **cozcontrol start** command is executed over the specified SSH connection to the Dataset Pipes SSH server subsystem on the target server. The Dataset Pipes subsystem starts a socket server to listen and process commands. Subsequent commands (fromdsn, todsn, cozclient, etc) retrieve connection properties from **cozcontrol**, use this information to establish a socket connection to the socket server, and execute the command on the server. The durable session ends when the **cozcontrol stop** command is issued (or an inactivity timeout occurs on the server) causing the socket server to shutdown and ending the SSH connection.

Alternatively, the **cozcontrol** command can be started with a tunneled option (`-t`). In this case, the initial SSH connection is configured as a control master with local port forwarding. Subsequent Dataset Pipes commands are forwarded to the server over the SSH connection established by **cozcontrol start**.

In both modes, the SSH connection remains established until either a **cozcontrol stop** command is executed or an inactivity timeout on the server occurs.

A user may set up concurrent durable sessions to multiple user/host pairs; however, not multiple sessions to the same user/host pair. To identify which durable session to use, set the environment variable `COZ_CONTROL_SESSION` to `user@host` prior to executing Dataset Pipes and **cozcontrol stop** commands.

Options

`start`
: Starts a durable connection to the Dataset Pipes subsystem on a target server.

  The **cozcontrol start** process runs in the background until a **stop** command is issued. If the terminal running the **start** command ends, the cozcontrol **start** process and its child SSH process will continue to run until the timeout occurs in the z/OS dspipes subsystem. This timeout value is configurable in `dspipes_config`.

`-t`
: Specifies that the **cozcontrol** durable connection be tunneled. In this case, the initial SSH connection is configured as a control master. Subsequent Dataset Pipes commands are forwarded over the SSH connection. The `-t` option is not set by default. When set, the environment variable `COZ_CONTROL_PATH` sets the control path. By default, the control path is `~/.ssh/cm-%r@%h:%p.sock`. This option can only be specified after

the **start** keyword.

Notes:

- OpenSSH 5.6 or later is required.
- This option is not supported on Windows using cygwin and OpenSSH. See *CYGWIN_controlMaster connections don't work* for the latest on this issue.
- If the target SSH server is not running on port 22, either EXPORT `COZ_CONTROL_PATH=~/.ssh/cm-%r@%h.sock` (i.e. removing `:%p` from the default value) or add the port configuration for the target host in `~/.ssh/config`. This issue will be fixed in a future release.

`-ssh` [ssh-options...] `[user]@host`
Specifies the ssh connection to the given z/OS user@host for the **cozcontrol start** command. This option is required and only permitted with the **start** keyword.

`stop`
Stops a durable connection to the Dataset Pipes subsystem on a target server.

`-h`
display help and exit.

`-v`
display the current version and exit.

Examples

Execute Dataset Pipes commands in a durable session

```
cozcontrol start -ssh user@host

fromdsn 'hlq.dsn' > /home/user/mydata/data1.txt

todsn 'hlq.dsn' < /home/user/mydata/data2.txt

cozclient wto "message to console"

cozcontrol stop
```

This example shows a durable session started for `user@host`. The next three commands (**fromdsn**, **todsn**, and **cozclient**) are executed over the socket connection established by **cozcontrol start**. Once the commands are complete, **cozcontrol stop** causes the socket connection to shutdown and the SSH connection to end.

Start a tunneled durable session

```
cozcontrol start -t -ssh user@host
```

Start a tunneled durable session for `user@host`. Subsequent Dataset Pipes commands are forwarded over the shared SSH connection established by **cozcontrol start**.

Name

fromdsn — write the contents of a z/OS dataset to stdout

Synopsis

```
fromdsn [OPTION...] dataset-name
fromdsn -ssh [ssh-opt...] [user]@host [OPTION...] dataset-name
fromdsn -local [OPTION...] dataset-name
fromdsn -v
fromdsn -h
```

Description

The **fromdsn** command reads a z/OS MVS dataset and writes a stream of data to stdout. Lines (if requested) are produced from dataset records based on the options provided.

The **fromdsn** command runs in one of three environments:

- locally (default on z/OS sytems)
- remotely, from a client which was started by Co:Z launcher.
- remotely, from a client that started a durable session to the server using the **cozcontrol** command.
- remotely, from a client-initiated ssh connection: `-ssh` option

The user has wide flexibility in choosing:

- How `dataset-name` is to be allocated/opened for writing
- How records are to be created from the incoming source lines
- What character set (codepage) translations are to be performed

`dataset-name` is automatically converted to upper case, and is assumed to be fully qualified unless otherwise specified (see the `-r` option below). If `dataset-name` starts with 'DD:', then it refers to an existing DDNAME.

The fromdsn command also supports reading JES spool files using special `dataset-name` syntax:

- `-JES.jobid` - reads the concatenated spool files for a given job. The `-S` option may be specified to indicate that SYSIN spool files should be included.
- `-JES.jobid.dsid` - reads a specific spool file by numerid dsid.
- `-JES.jobid.[stepname[.procstep]ddame` - reads the first spool file in a job that matches a step/procstep/ddname. This form may also be used to retrieve SYSIN spool files. When reading `-JES.jobid.JESJCLIN`, the output will include not only JCL card images, but also embedded SYSIN spool files.

Options

`-ssh` [ssh-options...] `[user]@host`
Specifies a remote invocation of **fromdsn** using a client-initiated ssh connection to the given z/OS user@host. If specified, this must be the first command option.

`-local`
Specifies the use of local z/OS I/O, even if run via CoZLauncher. Applicable when the source and target are both z/OS. If specified, this must be the first command option.

`-b`
binary mode, same as `-l` none `-p` 0x00.

`-h`
display help and exit.

`-k`
keep trailing pad characters in record. The default is to trim if `dataset-name` has fixed length records.

`-K`
always trim trailing pad characters, even if the dataset contains variable-length records.

`-l` line-separator
`nl` | `cr` | `lf` | `crlf` | `crnl`

follow lines with a newline, carriage return, linefeed, or combination. The characters are taken from the target codepage. The default is `nl`.

`rdw`

preceed lines with a four byte IBM-style RDW, consisting of a two byte network order (big endian) length, followed by two bytes of zeros.

`l4`

preceed lines with a four byte network order (big endian) length of the record that follows. Note: Unlike the `rdw` option, this length value does **not** include the size of the length field.

`mfrdw`

Write a 128 byte MicroFocus standard header prior to output data. Preceed each line with a network order (big endian) length. If the maximum record length is < 4095 bytes, the length field is 2 bytes. If the maximum record length is >= 4095 bytes, the length field is 4 bytes. Each line is padded with zeros to the nearest 4 byte boundary.

`0xbb[bb..]`

follow lines with a hex character sequence. The sequence must be between 1 and 8 bytes long.

`none`

no line separator

`-L` logging-options
: A comma-separated list of options to control logging and tracing.

  `M|A|C|E|W|N|I|D|T|F`

  Logging threshhold: eMergency, Alert, Critical, Error, Warning, Notice (default), Info, Debug, Trace, Fine.

  `t`

  Prefix log messages with a system timestamp

  `e`

  Include consumed cpu time in log messages

  `f=filename`

  Messages are logged to `filename` on the server instead of stderr. If not fully qualified, the file is written to the user's home directory on the server.

  `s`

  Messages are logged to SYSLOG facility instead of stderr

  `component=M|A|C|E|W|N|I|D|T|F`

  Set the logging threshold for a specific component. Specify only at the request of product support personnel.

`-o` fopen-options
: additional mode arguments to the z/OS C library fopen() routine. The base mode options used by **fromdsn** to open `dataset-name` are `rb,type=record,noseek"`. See "z/OS C++ Programming Guide" for details.

`-p` 0xbb
: pad character.

`-q` technique-str
: Codepage conversion technique string. Used to override the default Unicode Services value of `LMREC`. For more information, see IBM's Unicode Services User's Guide and Reference (SA22-7649).

`-r`
: `dataset-name` will be prefixed with the current z/OS userid.

`-s` source-codepage
: The codepage name or numeric CCSID id of the input dataset. If not specified, then the default z/OS process codepage is used.

`-S`
: This option may be specified when reading the concatenated spool files for a JES job ('-JES.jobid') to specify that the SYSIN spool files should also be included. This feature is only available on z/OS 1.10 or later.

`-t` target-codepage
: The codepage name or numeric CCSID id of data written to stdout. If not specified and invoked from a remote client with a line- separater other than 'none', 'rdw' or 'mfrdw', then the default client codepage is used,

otherwise the default z/OS code- page is used. Translation is disabled if source-codepage equals target-codepage.

`-T` STANDARD | translate_table_dsname
: Specifies the translate table to use for text mode transfers. This option overrides the `-s -t -q` options if also given. If `STANDARD`, the translate table TCPIP.STANDARD.TCPXLBIN is used. If a dataset name is supplied, it is expected to be in the format produced by the TSO CONVXLAT command. Only single byte translations are supported. Specifically, the dataset DCB must be LRECL=256,RECFM=F and contain two translation table records. The first record is an ASCII-to-EBCDIC mapping; the second record is an EBCDIC-to-ASCII mapping. Additional comment records (starting with * in the first column) are allowed.

`-v`
: display the current version and exit.

`-x` bpxwdyn-alloc-keywords
: can be specified to provide more precise control over the disposition of dataset-name than the fopen-options. For example, opening a dataset with fopen forces a disposition of 'OLD'. This may not always be desirable in a shared batch environment. The bpxwdyn keywords enable different dispostions to be specified (e.g 'SHR'). If `dataset-name` is 'DD:name', then this option is ignored. For a complete list of options, see the IBM manual: "Using REXX and z/OS UNIX System Services".

Files

fromdsn may obtain name matched profile information for a dataset from either a per-user profile or a system-wide profile on the z/OS system. For well known `dataset-name` patterns, profile options can be used to significantly reduce the specification of individual options on the command line. The file format and profile options are described in dsn_profile(5).

Examples

Local z/OS Examples

`fromdsn mvs1.my.lib(member1) > /home/user/member1`
: Copies an MVS dataset (PDS member) to an HFS/zFS file.

`fromdsn -x shr mvs1.input.dataset > /home/user/mydata`
: Copies an MVS dataset using DISP=SHR.

`fromdsn mvs1.input.dataset | todsn mvs1.output.dataset`
: Copies one MVS dataset to another

`fromdsn -jes.job123 > job.out`
: Copies all output from a job to an HFS/zFS file

`fromdsn -jes.j333.report.sysprint > report.txt`
: Copies the output from a job's spool file to an HFS/zFS file

Remote Client SSH Connection Examples

`fromdsn -ssh user@myzos2.com //mvs1.input.dataset > /tmp/data`
: Downloads an MVS dataset over an SSH connection (Unix).

```
fromdsn -ssh user@myzos2.com //mvs1.input.dataset > c:ata.txt
```

Downloads an MVS dataset over an SSH connection (Windows).

```
fromdsn -ssh -p 2222 user@myzos2.com -l rdw -r //binary.dataset > /tmp/rdw.bin.data
```

Downloads a MVS dataset over an SSH connection with addtional ssh options: (the dataset contains binary records which are prefixed with RDWs)

See Also

todsn(1)

Name

fromfile — write the contents of a z/OS POSIX file to stdout

Synopsis

```
fromfile [OPTION...] filename
fromfile -ssh [ssh-opt...] [user]@host [OPTION...] filename
fromfile -local filename
fromfile -v
fromfile -h
```

Description

The **fromfile** command reads a z/OS POSIX file and writes a stream of data to stdout. The produced stream of bytes are translated and given target system line terminators (if requested).

The **fromfile** command runs in one of three environments:

- locally (default on z/OS sytems)
- remotely, from a client which was started by Co:Z launcher.
- remotely, from a client that started a durable session to the server using the **cozcontrol** command.
- remotely, from a client-initiated ssh connection: `-ssh` option

`filename` is a path to the z/OS POSIX file to read. It may be either an absolute or relative path.

Options

`-ssh` [ssh-options...] `[user]@host`
: Specifies a remote invocation of **fromfile** using a client-initiated ssh connection to the given z/OS user@host. If specified, this must be the first command option.

`-local`
: Specifies the use of local z/OS I/O, even if run via CoZLauncher. Applicable when the source and target are both z/OS. If specified, this must be the first command option.

`-b`
: binary mode. Bytes are streamed as-is from the POSIX file to stdout.

`-h`
: display help and exit.

`-l` line-separator
`nl` | `cr` | `lf` | `crlf` | `crnl`

: follow lines with a newline, carriage return, linefeed, or combination. The characters are taken from the target

codepage. The default is `nl`.

`0xbb[bb..]`

follow lines with a hex character sequence. The sequence must be between 1 and 8 bytes long.

`none`

no line separator

`-L` logging-options
A comma-separated list of options to control logging and tracing.

`M|A|C|E|W|N|I|D|T|F`

Logging threshhold: eMergency, Alert, Critical, Error, Warning, Notice (default), Info, Debug, Trace, Fine.

`t`

Prefix log messages with a system timestamp

`e`

Include consumed cpu time in log messages

`f=filename`

Messages are logged to `filename` on the server instead of stderr. If not fully qualified, the file is written to the user's home directory on the server.

`s`

Messages are logged to SYSLOG facility instead of stderr

`component=M|A|C|E|W|N|I|D|T|F`

Set the logging threshold for a specific component. Specify only at the request of product support personnel.

`-q` technique-str
Codepage conversion technique string. Used to override the default Unicode Services value of `LMREC`. For more information, see IBM's Unicode Services User's Guide and Reference (SA22-7649).

`-s` source-codepage
The codepage name or numeric CCSID id of filename. If not specified, then the default z/OS process codepage is used.

`-t` target-codepage
The codepage name or numeric CCSID id of data written to stdout. If not specified and invoked from a remote client, the default client codepage is used. Translation is disabled if source-codepage equals target-codepage.

`-T` STANDARD | translate_table_dsname
Specifies the translate table to use for text mode transfers. This option overrides the `-s -t -q` options if also given. If `STANDARD`, the translate table TCPIP.STANDARD.TCPXLBIN is used. If a dataset name is supplied, it is expected to be in the format produced by the TSO CONVXLAT command. Only single byte translations

are supported. Specifically, the dataset DCB must be LRECL=256,RECFM=F and contain two translation table records. The first record is an ASCII-to-EBCDIC mapping; the second record is an EBCDIC-to-ASCII mapping. Additional comment records (starting with * in the first column) are allowed.

`-v`
: display the current version and exit.

Examples

Local z/OS Examples

`fromfile -b /etc/profile > /home/user/profile`
: Copies a file "as-is" to another location.

`fromfile -t ISO8859-1 myfile.txt > myfile_win.txt`
: Translates a file to the ISO8859-1 codepage from the default z/OS process codepage (e.g. IBM-1047).

Remote Client SSH Connection Examples

`fromfile -ssh user@myzos2.com -b /home/user/data.bin > /tmp/data.bin`
: Downloads binary data from z/OS to a remote system over over an SSH connection. No translation is performed.

`fromfile -ssh user@myzos2.com -t ISO8859-1 /etc/profile -l crlf > c:/mydir/profile.txt`
: Downloads a z/OS POSIX file over an SSH connection translating to a different code page and with Windows friendly line separators.

See Also

tofile(1)

Name

toasa — read a stream of data from stdin converting ASCII form-feed characters to ASA carriage control characters in stdout

Synopsis

```
toasa
toasa -v
toasa -h
```

Description

The **toasa** command converts ASCII form feeds to ASA control characters in a stream of data read from stdin. The converted output is written to stdout. Output lines will only have '1' (page eject) or ' ' (single line) carriage control in column one of each output line.

Options

`-h`
: display help and exit.

`-v`
: display the current version and exit.

Examples

Remote Client SSH Connection Examples

`cat /tmp/data | toasa | todsn -ssh user@myzos2.com -r //my.dataset`
: Uploads a file to an MVS Dataset over an SSH connection (Unix), converting the stream from /tmp/data to ASA format so that it is suitable for a z/OS dataset with RECFM=FBA.

Name

todsn — read a stream of data from stdin and write records to a z/OS dataset

Synopsis

```
todsn [OPTION...] dataset-name
todsn -ssh [ssh-opt...] [user]@host [OPTION...] dataset-name
todsn -local [OPTION...] dataset-name
todsn -v
todsn -h
```

Description

The **todsn** command writes records to `dataset-name` using a stream of data read from stdin. Dataset records are created from the input stream based on the options provided.

The **todsn** command runs in one of three environments:

- locally (default on z/OS sytems)
- remotely, from a client which was started by Co:Z launcher.
- remotely, from a client that started a durable session to the server using the **cozcontrol** command.
- remotely, from a client-initiated ssh connection: `-ssh` option

The user has wide flexibility in choosing:

- How `dataset-name` is to be allocated/opened for writing
- How records are to be created from the incoming source lines
- What character set (codepage) translations are to be performed

`dataset-name` is automatically converted to upper case, and is assumed to be fully qualified unless otherwise specified (see the `-r` option below). If `dataset-name` starts with 'DD:', then it refers to an existing DDNAME.

If `dataset-name` is `//INTRDR`, then the system internal reader is automatically allocated with a default RECFM=F and LRECL=80. In this case, the pseudo BPXWDYN option `symlist()` may be specified in order to pass one, or more JES system symbols to the internal reader. See *this example*.

Options

`-ssh` [ssh-options...] `[user]@host`
: Specifies a remote invocation of **todsn** using a client-initiated ssh connection to the given z/OS user@host. If specified, this must be the first command option.

`-local`

Specifies the use of local z/OS I/O, even if run via CoZLauncher. Applicable when the source and target are both z/OS. If specified, this must be the first command option.

`-a`
open `dataset-name` in append/mod mode. This option changes the base fopen() options to `ab,type=record,noseek`.

`-b`
binary flow mode, same as `-l` none `-p` 0x00 `-w` flow.

`-f`
begin writing data to `dataset-name` immediately (disable buffering). This is the default if used with a tunneled socket or if `dataset-name` refers to a SYSOUT data set.

`-h`
display help and exit.

`-l` line-separator
`flexible`|`cr`|`lf`|`crlf`|`nl`|`crnl`

source lines are separated by combination of linefeed and/or carriage return characters. The default is 'flexible' which allows for any of the other patterns above. These characters are taken from the source codepage.

`rdw`

source lines are preceeded with a four byte IBM-style RDW, consisting of a two byte network order (big endian) length followed by two bytes of zeros.

`mfrdw`

Source data is preceeded by a 128 byte MicroFocus standard header. Source lines are preceeded with a network order (big endian) length. If the maximum record length is < 4095 bytes, the length field is 2 bytes. If the maximum record length is >= 4095 bytes, the length field is 4 bytes. Each record must be padded with zeros to the nearest 4 byte boundary.

`0xbb[bb..]`

source lines are followed with a hex character sequence. The sequence must be between 1 and 8 bytes long.

`none`

source lines do not have separators; source lines are determined by the maximum output record length.

`-L` logging-options
A comma-separated list of options to control logging and tracing.

`M`|`A`|`C`|`E`|`W`|`N`|`I`|`D`|`T`|`F`

Logging threshhold: eMergency, Alert, Critical, Error, Warning, Notice (default), Info, Debug, Trace, Fine.

`t`

Prefix log messages with a system timestamp

`e`

Include consumed cpu time in log messages

`f=filename`

Messages are logged to `filename` on the server instead of stderr. If not fully qualified, the file is written to the user's home directory on the server.

`s`

Messages are logged to SYSLOG facility instead of stderr

`component=M|A|C|E|W|N|I|D|T|F`

Set the logging threshold for a specific component. Specify only at the request of product support personnel.

`-o` fopen-options
additional mode arguments to the z/OS C library fopen() routine. The base mode options used by **todsn** to open `dataset-name` are "wb,type=record,noseek". See "z/OS C++ Programming Guide" for details.

`-p` 0xbb
pad character. The default is the source codepage space character.

`-q` technique-str
Codepage conversion technique string. Used to override the default Unicode Services value of `LMREC`. For more information, see IBM's Unicode Services User's Guide and Reference (SA22-7649).

`-r`
`dataset-name` will be prefixed with the current z/OS userid.

`-s` source-codepage
The codepage name or numeric CCSID id of the input data. If not specified and invoked from a remote client with a line-separator other than 'none', 'rdw' or 'mfrdw', then the default client codepage is used, otherwise the default z/OS codepage is used.

`-t` target-codepage
The codepage name or numeric CCSID id of output dataset. If not specified, then the default z/OS process codepage is used. Translation is disabled if source-codepage equals target-code- page.

`-T` STANDARD | translate_table_dsname
Specifies the translate table to use for text mode transfers. This option overrides the `-s -t -q` options if also given. If `STANDARD`, the translate table TCPIP.STANDARD.TCPXLBIN is used. If a dataset name is supplied, it is expected to be in the format produced by the TSO CONVXLAT command. Only single byte translations are supported. Specifically, the dataset DCB must be LRECL=256,RECFM=F and contain two translation table records. The first record is an ASCII-to-EBCDIC mapping; the second record is an EBCDIC-to-ASCII mapping. Additional comment records (starting with * in the first column) are allowed.

`-v`
display the current version and exit.

`-w` wrap-options

`error`

an error is returned if the source line is longer than the maximum record length.

`flow`

source lines longer than the maximum record length are flowed across subsequent records. For fixed record formats, the pad character is used to complete the final record resulting from the source line.

`trunc`

source lines longer than the maximum record length are truncated

`wrap`

source lines longer than the maximum record length are broken into multiple records. The default is 'wrap'.

`-x` bpxwdyn-alloc-keywords
can be specified to provide more precise control over dataset allocation than the fopen-options. These allocation options allow `dataset-name` to be created with specific space and disposition parameters, or allow `dataset-name` to be created like an already existing dataset. If `dataset-name` is 'DD:name', then this option is ignored. For a complete list of options, see the IBM manual: "Using REXX and z/OS UNIX System Services".

`-z`
allow for an empty input stream. If not specified, the default is to exit with an error and not open or write to the output dataset if the input stream is empty.

Files

todsn may obtain name matched profile information for a dataset from either a per-user profile or a system-wide profile on the z/OS system. For well known `dataset-name` patterns, profile options can be used to significantly reduce the specification of individual options on the command line. The file format and profile options are described in dsn_profile(5).

Examples

Local z/OS Examples

```
todsn //MVS1.DATASET1 < myfile
```
Copies an HFS or zFS file to an MVS dataset.

```
todsn -o 'recfm=fb,lrecl=80' //MVS1.DATASET1 < myfile
```
Copies to an MVS dataset, overriding target DCB attributes.

```
todsn -w trunc //MVS1.DATASET1 < myfile
```
Copies to an MVS dataset, truncating long lines

```
todsn -x shr '//MVS1.MYLIB.DATA(MEMBER1)' < myfile
```
Copies to a PDS member, allocating with DISP=SHR.

```
todsn -r //test.data < myfile
```

Specifies a relative dataset name (HLQ will be added).

```
cat /home/user/ascii.txt | todsn -s iso8859-1 -r //my.dataset
```
Copies an ASCII HFS file to an EBCDIC MVS dataset.

```
cat /home/user/rdw.bin | todsn -l rdw -r //my.dataset
```
Copies a binary HFS file with RDW-prefixed lines to an MVS dataset.

```
todsn -x "symlist(*)" //intrdr <myjcl.txt
```
Submits a file as a job to the internal reader. The pseudo BPXWDYN keyword `symlist` is used to specify that all JES system symbols are to be passed to the internal reader (requires z/OS 2.1).

Remote Client SSH Connection Examples

```
todsn -ssh user@myzos2.com -r //my.dataset </tmp/myfile
```
Uploads a file to an MVS Dataset over an SSH connection (Unix).

```
copy c:ata.txt con: | todsn -ssh user@myzos2.com -r //my.dataset
```
Uploads a file to an MVS Dataset over an SSH connection (Windows).

```
todsn -ssh user@myzos2.com -r '//my.pds(mem1)' <myfile
```
Uploads a file to an MVS PDS Member over an SSH connection (Unix).

```
copy c:ata.txt con: | todsn -ssh user@myzos2.com -r '//my.pds(mem1)'
```
Upload a file to an MVS PDS Member over an SSH connection (Windows).

```
cat /tmp/data | todsn -ssh -p 2222 user@myzos2.com -r '//my.pds(mem1)'
```
Uploads a file to an MVS Dataset with additional ssh options.

See Also

fromdsn(1)

Name

tofile — read a stream of data from stdin and write to a z/OS POSIX file

Synopsis

```
tofile [OPTION...] filename
tofile -ssh [ssh-opt...] [user]@host [OPTION...] filename
tofile -local [OPTION...] filename
tofile -v
tofile -h
```

Description

The **tofile** command writes a stream of bytes to `filename` using a stream of data read from stdin. Codepage translation is performed and custom source line terminators are respected depending on the options provided.

The **tofile** command runs in one of three environments:

- locally (default on z/OS sytems)
- remotely, from a client which was started by Co:Z launcher.
- remotely, from a client that started a durable session to the server using the **cozcontrol** command.
- remotely, from a client-initiated ssh connection: `-ssh` option

`filename` is a path to the z/OS POSIX file to write. It may be either an absolute path or relative path.

Options

`-ssh` [ssh-options...] `[user]@host`
: Specifies a remote invocation of **tofile** using a client-initiated ssh connection to the given z/OS user@host. If specified, this must be the first command option.

`-local`
: Specifies the use of local z/OS I/O, even if run via CoZLauncher. Applicable when the source and target are both z/OS. If specified, this must be the first command option.

`-a`
: open `filename` in append mode.

`-b`
: binary mode. Bytes are streamed as-is from stdin to the POSIX file.

`-f`
: begin writing data to `filename` immediately (disable buffering). This is the default if used with a tunneled socket.

`-h`
: display help and exit.

`-l` line-separator
: `flexible|cr|lf|crlf|nl|crnl`

  source lines are separated by combination of linefeed and/or carriage return characters. The default is 'flexible' which allows for any of the other patterns above. These characters are taken from the source codepage.

  `0xbb[bb..]`

  source lines are followed with a hex character sequence. The sequence must be between 1 and 8 bytes long.

  `none`

  source lines do not have separators.

`-L` logging-options
: A comma-separated list of options to control logging and tracing.

  `M|A|C|E|W|N|I|D|T|F`

  Logging threshhold: eMergency, Alert, Critical, Error, Warning, Notice (default), Info, Debug, Trace, Fine.

  `t`

  Prefix log messages with a system timestamp

  `e`

  Include consumed cpu time in log messages

  `f=filename`

  Messages are logged to `filename` on the server instead of stderr. If not fully qualified, the file is written to the user's home directory on the server.

  `s`

  Messages are logged to SYSLOG facility instead of stderr

  `component=M|A|C|E|W|N|I|D|T|F`

  Set the logging threshold for a specific component. Specify only at the request of product support personnel.

`-m` file_access_mode
: the file access mode (as an octal number) to apply to `filename`.

`-n`
: do not replace `filename` if it exists.

`-p`
: make the path components to `filename` if they don't exist (ala **mkdir -p**).

`-q` technique-str
: Codepage conversion technique string. Used to override the default Unicode Services value of `LMREC`. For more information, see IBM's Unicode Services User's Guide and Reference (SA22-7649).

`-s` source-codepage
: the codepage name or numeric CCSID id of the data read from stdin. If not specified and invoked from a remote client, the default client codepage is used.

`-t` target-codepage
: the codepage name or numeric CCSID id of the output `filename`. If not specified, the default z/OS process codepage is used. Translation is disabled if source-codepage equals target-code- page.

`-T` STANDARD | translate_table_dsname
: Specifies the translate table to use for text mode transfers. This option overrides the `-s -t -q` options if also given. If `STANDARD`, the translate table TCPIP.STANDARD.TCPXLBIN is used. If a dataset name is supplied, it is expected to be in the format produced by the TSO CONVXLAT command. Only single byte translations are supported. Specifically, the dataset DCB must be LRECL=256,RECFM=F and contain two translation table records. The first record is an ASCII-to-EBCDIC mapping; the second record is an EBCDIC-to-ASCII mapping. Additional comment records (starting with * in the first column) are allowed.

`-u` umask
: the umask (as an octal number) to apply to `filename`.

`-v`
: display the current version and exit.

`-z`
: allow for an empty input stream. If not specified, the default is to exit with an error and not open or write to the output filename if the input stream is empty.

Examples

Local z/OS Examples

`tofile -t ISO8859-1 /home/user/myfile.iso8859 < myfile`
: Creates a copy of an HFS or zFS file locally, translating the default z/OS process codepage to ISO8859-1.

`tofile -p /home/user/newdir/myfile < myfile`
: Copies an HFS or zFS file to a new location, creating any missing path components (e.g. newdir).

Remote Client SSH Connection Examples

`tofile -ssh user@myzos2.com /home/user/mydata < /tmp/data`
: Uploads a remote file over an SSH connection (Unix). Codepage translation is performed from the remote unix codepage to the current z/OS process codepage.

`copy c:ata.txt con: | tofile -ssh user@myzos2.com myfile.txt`
: Uploads a remote file over an SSH connection (Windows). The target filename is relative to the current user's $HOME directory. Codepage translation is performed from the remote Windows codepage to the current z/OS process codepage.

```
tofile -ssh user@myzos2.com -b /home/user/data.bin < /tmp/data.bin
```

Uploads a remote file over an SSH connection as-is (no codepage translation is performed).

```
cat /tmp/myscript.sh | tofile -ssh -p 2222 user@myzos2.com -m 0777 /home/user/myscript.sh
```

Uploads a remote file with additional ssh options. The target file will be given a file access mask of 0777 (rwxrwxrwx), but is subject to the user's existing umask.

See Also

fromfile(1)

Appendix B. Command Reference - z/OS Utilities

- *catsearch(1)*
- *dsn_profile(5)*
- *jessym(1)*
- *lookupccsid(1)*
- *lsjes(1)*
- *pdsdir(1)*
- *safauth(1)*
- *saf-ssh-agent(1)*
- *showtrtab(1)*
- *wto(1)*
- *zsym(1)*

Name

catsearch — Co:Z utility to list z/OS catalogs

Synopsis

```
catsearch [-l] [-t [delim_char]] [-m max_entries] [-e entry_types] filter_key
catsearch [-x] [-e entry_types] filter_key
```

Description

This z/OS Co:Z utility command wraps the Catalog Search Interface (IGGCSI00) and provides a convenient display of information about the Datasets that match the supplied *`filter_key`*.

The syntax of the filter_key and additional documentation can be found in the following IBM publication: *DFSMS: Managing Catalogs - SC26-7409*.

Listing the entire catalog (filter_key `**`) is dis-allowed.

Options

`-l`
: Requests long form information about the listed Datasets. This information includes Volume, last referred date, tracks, used, recfm, lrecl, blocksize, dsorg and Dataset name.

`-t`
: Requests long form information about the listed Datasets in delimited format. If `delim_char` is supplied, it is used as a delimiter, otherwise a tab character (`\t`) is used.

`-m` max_entries
: Changes the maximum number of entries that will be returned by catsearch. the default is `2000`.

`-e` entry_types
: Changes the default entry type filter for catsearch. The default, if not specified, is `ABCGHRU`. Refer to *z/OS DFSMS Managing Catalogs: Catalog Search Interface* for more information..

`-x`
: Sets the exit code based on entries found. Entries found are not listed. With this option the following are ignored: `-l`, `-t`, and `-m`.

  The exit code is set according to the following:

  - 0 - no entries found
  - 1 - one entry found
  - 2 - more than one entry found

- 4 or greater - IGGCSI00 error (the return code)

Examples

1. This example shows a long listing `-l` form of a **catsearch**.

```
>catsearch -l user.coz.**
Volume  Referred  Ext  Tracks    Used Recfm Lrecl BlkSz Dsorg  Dsname
WORK81 2008/09/24   1      30       ?  U        0  6144  PO-E  USER.COZ.LOADLIB
WORK81 2008/09/24   1      15       4  FB      80 27920  PO    USER.COZ.SAMPJCL
WORK84 2008/09/11   1       1       1  U        0  6144  PS    USER.COZ.TEST.SEQ
WORK81 2008/09/24   1      15       4  FB      80 27920  PO    USER.COZ.TESTJCL
```

2. This example shows the difference between the single and double asterisk filter key symbols. A single asterisk only lists datasets within the current segment; the double asterisk will span segments.

```
>catsearch  -l user.coz.*
Volume  Referred  Ext  Tracks    Used Recfm Lrecl BlkSz Dsorg  Dsname
WORK81 2008/09/24   1      30       ?  U        0  6144  PO-E  USER.COZ.LOADLIB
WORK81 2008/09/24   1      15       4  FB      80 27920  PO    USER.COZ.SAMPJCL
WORK81 2008/09/24   1      15       4  FB      80 27920  PO    USER.COZ.TESTJCL

>catsearch  -l user.coz.**
Volume  Referred  Ext  Tracks    Used Recfm Lrecl BlkSz Dsorg  Dsname
WORK81 2008/09/24   1      30       ?  U        0  6144  PO-E  USER.COZ.LOADLIB
WORK81 2008/09/24   1      15       4  FB      80 27920  PO    USER.COZ.SAMPJCL
WORK84 2008/09/11   1       1       1  U        0  6144  PS    USER.COZ.TEST.SEQ
WORK81 2008/09/24   1      15       4  FB      80 27920  PO    USER.COZ.TESTJCL
```

3. Shows the use of the `-d` switch. Note that only the partial (pseudo directory) is listed for `USER.COZ.TEST`, and that there is no accompanying detailed information. Use of this option can be helpful when dealing with large catalogs.

```
>catsearch  -dl user.coz.**
Volume  Referred  Ext  Tracks    Used Recfm Lrecl BlkSz Dsorg  Dsname
WORK81 2008/09/24   1      30       ?  U        0  6144  PO-E  USER.COZ.LOADLIB
WORK81 2008/09/24   1      15       4  FB      80 27920  PO    USER.COZ.SAMPJCL
                                                              USER.COZ.TEST
WORK81 2008/09/24   1      15       4  FB      80 27920  PO    USER.COZ.TESTJCL
```

4. Shows the use of the `-x` switch. For illustration, the example below first shows a long listing using filter key `user.coz.*`. The result contains 3 datasets. The exit code using the `-x` switch and the same filter key is `2` indicating more than one entry found. The exit code is displayed by `echo $?`.

```
>catsearch  -l user.coz.*
Volume  Referred  Ext  Tracks    Used Recfm Lrecl BlkSz Dsorg  Dsname
WORK81 2008/09/24   1      30       ?  U        0  6144  PO-E  USER.COZ.LOADLIB
WORK81 2008/09/24   1      15       4  FB      80 27920  PO    USER.COZ.SAMPJCL
WORK81 2008/09/24   1      15       4  FB      80 27920  PO    USER.COZ.TESTJCL

>catsearch  -x user.coz.*
```

```
>echo $?
2
```

Name

dsn_profile — profile information for dataset-name patterns

Synopsis

```
/etc/dsn_profile
~/.dsn_profile
```

Description

todsn and **fromdsn** read `dataset-name` profile information from `/etc/dsn_profile`, or if present `~/.dsn_profile`. This file contains stanzas of the form:

```
program-name dataset-name-pattern
    keyword value
    keyword value
    ...
```

program-name must start in column 1 of the line and may be either **todsn** or **fromdsn**. Keyword value pairs are read until the start of a new stanza is encountered. Lines starting with '#' and empty lines are interpreted as comments.

`dataset-name-pattern` is a string conforming to the `fnmatch()` C library function pattern language.

The possible keywords and allowed values follow. Keywords are applicable to both **todsn** and **fromdsn** unless noted otherwise. Keywords and values are case-insensitive.

`allocKeywords` | `alloc`
: bpxydyn dataset allocation options. For a complete list of options, see "Using REXX and z/OS UNIX System Services".

`lineTerminationRule`
: `flexible` | `lf` | `cr` | `crlf` | `nl` | `crnl` | `rdw` | `mfrdw` | `0xbb[bb..]` | `none`.

  `flexible` is only applicable to **todsn**.

`openOptions` | `extraOpenOptions`
: Additional mode options to be added to the base options on the `fopen()` call.

`padChar`
: the pad character.

`recordOverflowRule`
: One of: `error` | `flow` | `trunc` | `wrap`. This keyword is not applicable to **fromdsn**.

`relative`
: the `dataset-name` supplied is relative, and the MVS userid will be added.

`sourceCodePage`
: the source character set.

`targetCodePage`
: the target character set.

`trim`
: trailing pad characters are trimmed.

Files

`/etc/dsn_profile`
: Contains system wide profile data for **fromdsn** and **todsn**.

`~/.dsn_profile`
: if present, will be read instead, allowing individual users to define their own profile data.

Examples

```
# Force dataset-name containing '.JCL' to be RECFM=FB and LRECL=80
todsn *.JCL*
     openOptions recfm=fb,lrecl=80

# Set the codepage and trim option for any dataset name ending with '.ASCII'
fromdsn *.ASCII
     targetCodePage ISO8859-1
     trim true
```

See Also

fromdsn(1), todsn(1)

Name

jessym — Command line interface to the JES Symbol Service

Synopsis

```
jessym name
jessym [-p prefix] -s name-pattern ...
jessym [-p prefix] -x name-pattern ...
jessym [-r] -c name=value ...
jessym -u name=value ...
jessym -d name-pattern ...
```

Description

This z/OS Co:Z utility uses the JES Symbol Service (IAZSYMBL) to extract, create, update, and delete JES system symbols. Requires z/OS 2.1 or later.

Options

`-p`
: Specifies a prefix to be added to JES Symbol names when using the (`-s`) or (`-x`) options.

`-s`
: Prints the value of one or more JES Symbols whose name matches a `name-pattern`. Characters in a name pattern are automatically folded to upper case and may include `*` or `?` characters to match zero-or-more or exactly-one characters respectively. If no name-patterns are given, then the default is `*` (all names). Each line is displayed on stdout in the form: NAME='VALUE'

`-x`
: Prints an export command with the value of one or more JES Symbols whose name matches a `name-pattern`. Characters in a name pattern are automatically folded to upper case and may include `*` or `?` characters to match zero-or-more or exactly-one characters respectively. If no name-patterns are given, then the default is `*` (all names). Each line is displayed on stdout in the form: export NAME='VALUE'

`-r`
: Specifies that when defining a new symbol (with option `-c`) that the value of an existing symbol of the same name may be replaced.

`-c`
: Creates one or more new symbols given arguments of the form: `NAME=VALUE`. Characters in the name (but not the value) are folded to uppercase automatically. If the `-r` is also specified, then the value of an existing symbol with the same name will be replaced. Symbols will be created at the job (address space) level.

`-u`
: Update one or more existing symbols with a new value given arguments of the form: `NAME=VALUE`. Characters in the name (but not the value) are folded to uppercase automatically. The symbols must previously exist; a new symbol will not be created.

`-d`

Delete one or more symbols that match the given name pattern(s). Characters in a name pattern are automatically folded to upper case and may include `*` or `?` characters to match zero-or-more or exactly-one characters respectively.

See Also

The **todsn** command has been enhanced for z/OS 2.1 to support passing JES symbols to jobs submitted to the internal reader.

The COZBATCH utility has been enhanced for z/OS 2.1 so that the values of all JES symbols will be automatically exported as environment variables with a prefix of `JES_`.

Examples

1. Create a new JES symbol and display it by name

```
> jessym -c A=B
> jessym A
B
```

2. Create or replace a JES symbol and display it

```
> jessym -r -c A=c
> jessym a   # symbol names are automatically folded to uppercase
c
```

3. Show symbols matching a name pattern

```
> jessym -s SYS*
SYS_CORR_CURRJOB='S0000434DTLZOS01CC27C5EA.......:                              '
```

4. Generate export statements for all symbols

```
> jessym -x   # defaults to * (all)
export SYS_CORR_CURRJOB='S0000434DTLZOS01CC27C5EA.......:                       '
export A='c'
```

5. Generate export statements for all symbols, using a name prefix

```
> jessym -p JES_ -x
export JES_SYS_CORR_CURRJOB='S0000434DTLZOS01CC27C5EA.......:                   '
export JES_A='c'
```

6. Generate export statements for all symbols, using a name prefix, and pipe these as commands into the current shell. Note that this is done automatically by the COZBATCH utility.

```
> set -o pipecurrent  # this shell option required to use the current shell
> jessym -p JES_ -x  | . /dev/fd0
> echo $JES_A
c
```

7. Display one symbol and read its value into a shell variable

```
> set -o pipecurrent  # this shell option required to use the current shell
> jessym A  | read myA
> echo $myA
c
```

8. Delete a symbol

```
> jessym -d A
> jessym A
JesSymbols[W]: IAZSYMBOL rc=0 RET=4 REAS=4
```

Name

lookupccsid — Co:Z utility to return the coded character set identifier (CCSID) associated with a character set

Synopsis

```
lookupccsid codesetName
```

Description

This z/OS Co:Z utility is useful for determining the unicode services CCSID associated with a character set.

This program uses the `__toCcsid()` z/OS C runtime library function to determine the numeric CCSID associated with `codesetName`. If unsuccessful, 0 is returned

Examples

```
/dovetail/coz/bin: > lookupccsid UTF-8
1208 UTF-8

/dovetail/coz/bin: > lookupccsid ISO8859-1
819 ISO8859-1
```

Name

lsjes — Co:Z utility to display JES job and spool file status

Synopsis

```
lsjes [-t [delim_char]] [-o userid] [-p jobname-pattern] [-s a|i|o]
lsjes [-t [delim_char]] -i jobid ...
lsjes [-t [delim_char]] [-S] -d jobid ...
```

Description

This z/OS Co:Z utility uses the Extended Status Subsystem Interface to query the status of jobs in the primary JES2 or JES3 subsystem.

The first form displays a list, one line per job, all jobs that match optional filter criteria. If no arguments are specified, then all jobs owned by the current userid are displayed.

The second form displays one or more specific jobs, along with their spool files.

Options

`-t`
: Requests output in delimited format. If `delim_char` is supplied, it is used as a delimiter, otherwise a tab character (`\t`) is used. If this option is used, then header lines are not displayed in the listing.

`-o` userid
: Filters the job listing to include only jobs whose owner is the given z/OS userid. If this option is omitted, then jobs are filtered using the current userid.

`-p` jobname-pattern
: Filters the job listing to include only jobs with a name matching the given pattern. Valid generic pattern characters include '*' and '%'.

`-s` a|i|o
: Filters the job listing to include only jobs whose status is either "ACTIVE", "INPUT", or "OUTPUT".

`-i`
: Filters the job listing to include only the job(s) specified. One or more jobids must follow, where each jobid is 2-8 characters that starts with one of the prefixes "J/JO/JOB/T/TS/TSU/S/ST/STC/I/IN/INT" followed by a number.

`-d`
: This option indicates the second form of the command (detail mode), in which specific jobs and their spool files are listed. One or more jobids must follow, where each jobid is 2-8 characters that starts with one of the prefixes "J/JO/JOB/T/TS/TSU/S/ST/STC/I/IN/INT" followed by a number.

`-S`

This option may be precede the `-d` option to indicate that the listing of spool files should also include SYSIN files, including JESJCLIN. This feature is only available on z/OS 1.10 or later.

See Also

The **fromdsn** can be used to read the contents of a job's spool files.

Examples

1. This example lists all jobs owned by the current userid.

```
>lsjes
Jobid    Jobname  Owner    Status   Class Completion
TSU02611 KIRK     KIRK     OUTPUT   TSU   ABEND=622
JOB02663 KIRKJ1   KIRK     OUTPUT   A     RC=0000
JOB02662 KIRKJ1   KIRK     OUTPUT   A     RC=0000
JOB02661 KIRKJ1   KIRK     OUTPUT   A     RC=0000
JOB02660 KIRKJ1   KIRK     OUTPUT   A     RC=0000
JOB02659 KIRKJ1   KIRK     OUTPUT   A     RC=0000
JOB02462 COZOOM   KIRK     OUTPUT   A     RC=0000
JOB02460 COZOOM   KIRK     OUTPUT   A     RC=0255
```

2. As above, but with delimiters (and without a header).

```
>lsjes -t'|'
TSU02611|KIRK|KIRK|OUTPUT|TSU|ABEND=622
JOB02663|KIRKJ1|KIRK|OUTPUT|A|RC=0000
JOB02662|KIRKJ1|KIRK|OUTPUT|A|RC=0000
JOB02661|KIRKJ1|KIRK|OUTPUT|A|RC=0000
JOB02660|KIRKJ1|KIRK|OUTPUT|A|RC=0000
JOB02659|KIRKJ1|KIRK|OUTPUT|A|RC=0000
JOB02462|COZOOM|KIRK|OUTPUT|A|RC=0000
JOB02460|COZOOM|KIRK|OUTPUT|A|RC=0255
JOB02447|COZOOM|KIRK|OUTPUT|A|RC=0255
JOB02446|COZOOM|KIRK|OUTPUT|A|RC=0255
JOB02334|KIRKSLP|KIRK|OUTPUT|A|RC=0000
JOB02333|KIRKSLP|KIRK|OUTPUT|A|RC=0000
JOB02332|KIRKSLP|KIRK|OUTPUT|A|RC=0000
JOB02331|KIRKSLP|KIRK|OUTPUT|A|RC=0000
JOB02306|KIRKSLP|KIRK|OUTPUT|A|RC=0000
JOB02123|KIRKCB|KIRK|OUTPUT|B|RC=0001
JOB02070|KIRKCT|KIRK|OUTPUT|A|RC=4000
```

3. Tabbed delimiters can used with the Unix **cut** to select a field:

```
>lsjes -t | cut -f1
TSU02611
JOB02663
JOB02662
JOB02661
JOB02660
JOB02659
JOB02462
JOB02460
JOB02447
```

```
JOB02446
JOB02334
JOB02333
JOB02332
JOB02331
JOB02306
JOB02123
JOB02070
```

4. This example lists all active jobs (any owner).

```
>lsjes  -o'*' -sa
Jobid    Jobname  Owner    Status   Class Completion
STC02691 BPXAS    OMVSKERN ACTIVE   STC
STC02689 BPXAS    OMVSKERN ACTIVE   STC
STC02688 BPXAS    OMVSKERN ACTIVE   STC
...
```

5. To list all jobs using a jobname pattern (any owner).

```
>lsjes -o '*' -p 'T*'
Jobid    Jobname  Owner    Status   Class Completion
STC02556 TCPIP    TCPIP    OUTPUT   STC   RC unknown
STC02579 TCAS     STRTASK  OUTPUT   STC   RC unknown
STC02093 TCPIP    TCPIP    OUTPUT   STC   -HELD-
STC02608 TCAS     STRTASK  ACTIVE   STC
STC02605 TN3270   TCPIP    ACTIVE   STC
STC02586 TCPIP    TCPIP    ACTIVE   STC
...
```

6. To display the status of a job and list its spool files:

```
>lsjes -d J2333
Jobid    Jobname  Owner    Status   Class Completion
JOB02333 KIRKSLP  KIRK     OUTPUT   A     RC=0000
         Id  Stepname Procstep DDName   C Owner    Recfm Lrecl Bytes
         002 JES2             JESMSGLG H KIRK     FA      133 1313
         003 JES2             JESJCL   H KIRK     V       136 253
         004 JES2             JESYSMSG H KIRK     VA      137 823
         102 UNIX             SYSOUT   H KIRK     FBA     121 428
```

Name

pdsdir — Co:Z utility to list Partitioned dataset members and their statistics, if available.

Synopsis

```
pdsdir [-n] hlq.dataset.name
```

Description

This z/OS Co:Z utility lists the members of the PDS *hlq.dataset.name* . If statistics are available, they are listed.

Options

`-n`
: Only member names are listed.

Examples

1. This example shows a PDS directory listing.

```
 >pdsdir user.coz.sampjcl
Name           Size  Created         Changed          ID
@@README
BPXBATCH         13 2008/04/04 2008/04/04 17:18:09  USER
BPXBATSL         16 2008/04/03 2008/04/03 10:36:52  USER
COZCFGD          65 2008/03/27 2008/05/12 14:28:54  USER
COZPROC          30 2008/03/27 2008/03/27 11:54:48  USER
DTLSPAWN         40 2008/05/05 2008/05/05 09:31:08  USER
GPGDSN           15 2008/05/05 2008/05/05 10:40:05  USER
GREPDSN
GREPSED          12 2008/05/05 2008/05/05 09:30:51  USER
OFFLDSMF
RUNCOZ           20 2008/03/27 2008/09/24 17:05:53  USER
RUNCOZ2          15 2008/05/05 2008/05/05 10:02:51  USER
RUNCOZ3           8 2008/05/05 2008/05/06 08:50:37  USER
RUNSPAWN         54 2008/05/12 2008/05/12 14:25:37  USER
RUNSPWN2         20 2008/05/12 2008/05/12 13:19:05  USER
TDIRK            18 2008/04/03 2008/04/03 10:19:20  USER
WGET2DSN
```

Name

safauth — Co:Z utility to check the current user's authorization to a SAF (RACF) resource.

Synopsis

```
safauth saf-class saf-entity [read | update | control | alter] [volser]
```

Description

This z/OS Co:Z utility is a wrapper for the RACROUTE REQUEST=AUTH macro and can be used to check the current user's access to a given SAF(RACF) resource.

An exit code of zero indicates that the auth check passed; otherwise the non-zero return code from the RACROUTE macro is returned as the exit code.

RACROUTE REQUEST=AUTH requires VOLSER= for CLASS=DATASET, but it is not used for SMS managed datasets. The `volser` option is ignored if `CLASS!=DATASET`, but if `volser` is not specified and CLASS=DATASET, then `volser` defaults to `DUMMY`.

Name

saf-ssh-agent — Co:Z utility to enable ssh client authetication via SAF/RACF Digital Certificates

Synopsis

```
saf-ssh-agent -x [-f export_file] keyring[:label]
saf-ssh-agent -b asn1_file  keyring[:label]
saf-ssh-agent -c keyring[:label] command [command_args...]
```

Description

This z/OS Co:Z utility is similar in function to the OpenSSH **ssh-agent**, but rather than automatically authenticating the ssh client with ssh keys, it provides for authentication with SAF/RACF Digital Certificates.

`keyring[:label]` is the keyring (and optional certifcate label) to use.

Options

`-x`
: extract the public key from a SAF/RACF Digital Certificate in OpenSSH format.

`-f` export_file
: The file to export the OpenSSH format key. If this option is omitted, the key will be written to `stdout`.

`-b` asn1-file
: extract the public key (in binary ASN1 format) to a file. This option is used for diagnostic purposes.

`-c`
: run *`command`* as a child process after initializing **saf-ssh-agent**. This enables *`command`* to authenticate with the supplied *`keyring[:label]`*. Generally, this option is used to run **ssh** as a child process, allowing it to take advantage of SAF RACDCERT authentication.

Examples

1. This example shows how to extract an OpenSSH public key from a SAF/RACF Digital Certificate. In this case, the key is written to `stdout`.

```
/dovetail/coz/bin: > saf-ssh-agent -x MY-RING

ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAAAgQDVoW8HzKQYIfVqOZpEHgPLLfUkqg68fyBc
XTDUpFyQiIoKWRh1rHHa4DlQxa80lMPzr+VvyzvJrgzXI0OVp9A09yLgr4XxtrkrfTY3nojT
35y3bZqZXTefCX5atN8yaORfkXZeYl4H+ojdQK3ywHdDlqOMTSl1Cj4/9w67JNTXXw== CN=
Stephen Goetze,OU=Development,O=Dovetailed Technologies,C=US
```

1. This example shows how to run ssh as a child process to execute the **who** command on the remote system

linux.com. The ssh client will authenticate via the SAF RACDCERT contained in `MY-RING`.

```
/dovetail/coz/bin: > saf-ssh-agent -c MY-RING ssh myid@linux.com who

myid   tty7         2009-12-29 06:15 (:0)
myid   pts/0        2009-12-29 11:23 (:0.0)
myid   pts/1        2010-01-08 11:43 (:0.0)
```

Name

showtrtab — Co:Z utility to display a translation table

Synopsis

```
showtrtab [-L logging_options][-s source_codepage][-t target_codepage][-q technique_str]
```

Description

This z/OS Co:Z utility command will show the translate table associated with a source and target codeset. It first attempts to use unicode services, but will fall back to `iconv()` if needed.

If a table cannot be built, the command will display error information that may be useful in determining the problem.

This utility only supports SBCS -> SBCS and SBCS -> MBCS. MBCS -> SBCS tables are not supported.

To get detailed information, the logging option `-LTranslator=T` can be used.

Options

`-L` logging-options
: A comma-separated list of options to control logging and tracing:

  `M|A|C|E|W|N|I|D|T`

  Logging threshhold: eMergency, Alert, Critical, Error, Warning, Notice, Info (default), Debug, Trace.

  `t`

  Prefix log messages with a system timestamp

  `e`

  Include consumed cpu time in log messages

  `s`

  Messages are logged to SYSLOG facility instead of stderr

  `logname=M|A|C|E|W|N|I|D|T`

  Set a specific log name to the given threshold

`-s` source-codepage
: The source codepage name. If not specified, then the default z/OS process codepage is used. At least one of `-s` or `-t` is required.

`-t` target-codepage

The target codepage name. If not specified, then the default z/OS process codepage is used. At least one of `-s` or `-t` is required.

`-q` technique-str
The Unicode Services conversion technique(s) to accept. This is a string of one or more of the following technique characters:

C
Customized Subset

E
Enforced Subset

L
Language Environment Behavior

M
Modified Language Environment Behavior

R
Roundtrip

If more than one character is specified, the first available matching technique is used - therefore the order is significant.

When falling back to `iconv()` this list is ignored

Examples

1. This example shows a Translate table from a source code page of ISO8859-1 to a target codepage which is the current z/OS process' default

```
/dovetail/coz104/bin: > showtrtab -s ISO8859-1

00:  00 01 02 03   37 2D 2E 2F   16 05 15 0B   0C 0D 0E 0F
10:  10 11 12 13   3C 3D 32 26   18 19 3F 27   1C 1D 1E 1F
20:  40 5A 7F 7B   5B 6C 50 7D   4D 5D 5C 4E   6B 60 4B 61
30:  F0 F1 F2 F3   F4 F5 F6 F7   F8 F9 7A 5E   4C 7E 6E 6F
40:  7C C1 C2 C3   C4 C5 C6 C7   C8 C9 D1 D2   D3 D4 D5 D6
50:  D7 D8 D9 E2   E3 E4 E5 E6   E7 E8 E9 AD   E0 BD 5F 6D
60:  79 81 82 83   84 85 86 87   88 89 91 92   93 94 95 96
70:  97 98 99 A2   A3 A4 A5 A6   A7 A8 A9 C0   4F D0 A1 07
80:  20 21 22 23   24 25 06 17   28 29 2A 2B   2C 09 0A 1B
90:  30 31 1A 33   34 35 36 08   38 39 3A 3B   04 14 3E FF
A0:  41 AA 4A B1   9F B2 6A B5   BB B4 9A 8A   B0 CA AF BC
B0:  90 8F EA FA   BE A0 B6 B3   9D DA 9B 8B   B7 B8 B9 AB
C0:  64 65 62 66   63 67 9E 68   74 71 72 73   78 75 76 77
D0:  AC 69 ED EE   EB EF EC BF   80 FD FE FB   FC BA AE 59
E0:  44 45 42 46   43 47 9C 48   54 51 52 53   58 55 56 57
F0:  8C 49 CD CE   CB CF CC E1   70 DD DE DB   DC 8D 8E DF
```

2. This example shows a Translate table from a source code page of ISO8859-2 to a target codepage of IBM-273. Logging is activated.

```
/dovetail/coz104/bin: > showtrtab -LTranslator=T -s ISO8859-2 -t IBM-273
showtrtab[T]: Translator: -> Translator(ISO8859-2, IBM-273, LMREC)
showtrtab[T]: Translator: -> getCodePage(ISO8859-2)
showtrtab[D]: Translator: Looking for codepage substitution environment
   variable: COZ_TRSUB_ISO8859-2
showtrtab[T]: Translator: <- getCodePage()
showtrtab[T]: Translator: -> getCodePage(IBM-273)
showtrtab[D]: Translator: Looking for codepage substitution environment
   variable: COZ_TRSUB_IBM-273
showtrtab[T]: Translator: <- getCodePage()
showtrtab[T]: Translator: -> initialize( ISO8859-2->IBM-273, t=LMREC)
showtrtab[T]: Translator: -> getCcsid(ISO8859-2)
showtrtab[T]: Translator: <- getCcsid(912)
showtrtab[T]: Translator: -> getCcsid(IBM-273)
showtrtab[T]: Translator: <- getCcsid(273)
showtrtab[T]: Translator: -> initCunbcprm()
showtrtab[T]: Translator: <- initCunbcprm()
showtrtab[T]: Translator: <- initialize()
showtrtab[T]: Translator: <- Translator()
00:  00 01 02 03   37 2D 2E 2F   16 05 25 0B   0C 0D 0E 0F
10:  10 11 12 13   3C 3D 32 26   18 19 3F 27   1C 1D 1E 1F
20:  40 4F 7F 7B   5B 6C 50 7D   4D 5D 5C 4E   6B 60 4B 61
30:  F0 F1 F2 F3   F4 F5 F6 F7   F8 F9 7A 5E   4C 7E 6E 6F
40:  B5 C1 C2 C3   C4 C5 C6 C7   C8 C9 D1 D2   D3 D4 D5 D6
50:  D7 D8 D9 E2   E3 E4 E5 E6   E7 E8 E9 63   EC FC 5F 6D
60:  79 81 82 83   84 85 86 87   88 89 91 92   93 94 95 96
70:  97 98 99 A2   A3 A4 A5 A6   A7 A8 A9 43   BB DC 59 07
80:  20 21 22 23   24 15 06 17   28 29 2A 2B   2C 09 0A 1B
90:  30 31 1A 33   34 35 36 08   38 39 3A 3B   04 14 3E FF
A0:  41 44 46 47   9F 49 52 7C   BD 54 57 58   64 CA 66 67
B0:  90 69 70 72   BE 74 77 78   9D 80 8A 8B   8C 8E 8F 9A
C0:  9B 65 62 9C   4A 9E A0 68   AA 71 AB 73   AE 75 76 AF
D0:  AC B0 B1 EE   EB B2 E0 BF   B3 B4 FE B6   5A AD B7 A1
E0:  B8 45 42 B9   C0 BA BC 48   CC 51 CD 53   CF 55 56 DA
F0:  DB DD DF CE   CB EA 6A E1   ED EF DE FA   D0 8D FB FD
```

3. Shows an attempt to build a MBCS->SBCS table, and the resulting error.

```
/dovetail/coz104/bin: > showtrtab -s UTF-8 -t IBM-1047
showtrtab[E]: TranslateException: Exception occurred during translation,
   RC=4, Reason=12
```

Name

wto — Co:Z utility to issue a Write To Operator (WTO) from USS.

Synopsis

```
wto [-r ROUTCDE,...] [-d DESC,...] message
```

Description

This z/OS Co:Z utility command issues *`message`* as a write to operator (WTO).

If the ROUTCDE or DESC codes are omitted, the system uses the routing code specified on the ROUTCODE keyword on the DEFAULT statement in the CONSOLxx member of SYS1.PARMLIB.

NOTE: The message will be prefixed by: `BPXM023I (userid)` unless the userid has access to "BPX.CONSOLE" in the SAF "FACILITY" class. Additionally, in order to prevent a recursive logging error, the **wto** command will fail with an error message when logging has been redirected to `/dev/console`.

Messages with embedded spaces must be quoted.

Options

`-r` ROUTCDE
: Specifies the routing code(s) for the message:

  `1` - Operator Action

  `2` - Operator Information

  `3` - Tape Pool

  `4` - Direct Access Pool

  `5` - Tape Library

  `6` - Disk Library

  `7` - Unit Record Pool

  `8` - Teleprocessing Control

  `9` - System Security

  `10` - System/Error Maintenance

  `11` - Programmer Information

  `12` - Emulation

`13-128` - See *MVS Programming: Authorized Assembler Services Reference, Volume 4 (SETFRR-WTOR) - SA22-7612*

`-d` DESCR
Specifies the descriptor(s) for the message:

`1` - System Failure (*)

`2` - Immediate Action Required (*)

`3` - Eventual Action Required (*)

`4` - System Status (*)

`5` - Immediate Command Response (*)

`6` - Job Status (*)

`7` - Task-Related

`8` - Out-of-Line

`9` - Operator's Request

`10` - Not Defined

`11` - Critical Eventual Action Required (*)

`12` - Important Information (*)

(*) Mutually exclusive

Examples

1. This example shows a WTO, using ROUTCDE "Programmer Information" and DESCR "Important Information".

```
>wto -r 11 -d 12 "status message"
```

Name

zsym — Co:Z utility to list system symbol values.

Synopsis

```
zsym "&symbol"
```

Description

This z/OS Co:Z utility lists the value of *symbol*. Note that the symbol must be preceeded by an ampersand (&) and enclosed in quotes.

Examples

1. Show various system symbol values

```
>zsym "&SYSNAME"
S0W1
>zsym "&SYSPLEX"
SVSCPLEX
>zsym "&YYMMDD"
080925
```

Appendix C. Co:Z Environment Variables

The following table describes the environment variables defined by the Co:Z Toolkit. These variables can be set to override default behavior.

Table C.1. Miscellaneous options

Variable	Description
COZ_SSH_CMD	Specifies an alternate executable for the SSH client used to connect to z/OS. By default, this is `ssh`. For example, to use the PuTTY command line client `plink` instead of `ssh` set `COZ_SSH_CMD=/path/to/plink`.
COZ_SSH_OPTS	Convenience setting for supplying SSH options, including userid and host when making remote dataset pipes calls. For example, the command `fromdsn -ssh user@host //mydsn` can be simplified to `fromdsn //mydsn` if `COZ_SSH_OPTS` is set to `user@host`. This is very handy for repeated use of the remote dataset pipes commands. When this varaible is set, the **cozcontrol** command can be simplified to **cozcontrol start** (omitting the -ssh user@host parameters). When a durable session is active, subsequent dataset pipes commands ignore the environment variable setting and use the durable session.
COZ_SSH_SUBSYS	Specifies an alternate SSH server subsystem name for Dataset Pipes. By default, this is `dspipes`.
COZ_CLIENT_CODEPAGE	Changes the default client code page, which is used for codepage translation in text mode data transfers (i.e. if the `-t` is not supplied). By default, the default client code page is set the result of the POSIX system call `nl_langinfo(CODESET)`.
COZ_DEFAULT_LOGSTREAM	Changes the default stream that the Co:Z Log facility writes its messages to. By default, this is the `stderr` stream.
COZ_LOG	Sets log level for CozServer session level logging. The default is N, Notice.
COZ_LOG_CMD	Sets logging level for Dataset Pipes commands running on the server (fromdsn, cozclient, etc). The default is N, Notice. Command tracing can alternately be enabled with the -L option on most Dataset Pipes commands.
COZ_LOG_CMD_DUP	When set to true (default is false), duplicates tracing enabled by COZ_LOG_CMD to the session log. This is recommended when

Variable	Description
	requesting support from Co:Z support personnel because all logging for a problem is captured in a single file.
DSPIPES_LOGFILE	Pathname of file to where DSPIPES log/debug messages are written. The default is `/tmp/dspipes.<userid>.<...>.log`
DSPIPES_LOGDIR	Directory name (without trailing slash) where DSPIPES log files are created, rather than /tmp or $TMPDIR. This variable is ignored if `DSPIPES_LOGFILE` is set.
COZ_CONTROL_SESSION	Used to identify the `user@host` associated with a `cozcontrol` durable connection. This setting is required when a user has multiple concurrent durable connections. The setting identifies the connection to use for remote Dataset Pipes commands as well as `cozcontrol stop`.
COZ_CONTROL_PATH	Used to override the default control path created with a tunneled `cozcontrol` durable connection. The default, when this variable is not set, is `~/.ssh/cm-%r@%h:%p.sock`.

Appendix D. License

The Co:Z Co-Processing Toolkit for z/OS, comprised of Co:Z Launcher, Co:Z Dataset Pipes, Co:Z SFTP, Co:Z Batch, Co:Z ssh-proxyc and Co:Z Target System Toolkit (in object code form only) is distributed under the Co:Z Community License Agreement (see below). *Note:* This community license is superseded for Co:Z Toolkit Enterprise License and Support customers. All components are distributed in binary form.

```
Co:Z COMMUNITY LICENSE AGREEMENT

PLEASE READ THIS COMMUNITY LICENSE AGREEMENT (THIS “AGREEMENT”) CAREFULLY.
THIS AGREEMENT SETS FORTH THE TERMS ON WHICH DOVETAILED TECHNOLOGIES, LLC
(“DOVETAIL”), A MISSOURI LIMITED LIABILITY COMPANY, MAKES AVAILABLE THE
CO:Z CO-PROCESSING TOOLKIT FOR z/OS AT NO CHARGE FOR DOWNLOAD, INSTALLATION AND
USE BY THE COMMUNITY.  BY DOWNLOADING, INSTALLING, OR USING THE SOFTWARE, YOU
ACKNOWLEDGE THAT YOU HAVE READ, UNDERSTAND, AND AGREE TO BE LEGALLY BOUND BY
THIS AGREEMENT.

1.  DEFINITIONS.  As used in this Agreement, the following capitalized terms
shall have the following meanings:

“Documentation” means Dovetail’s accompanying user documentation for the
Software, as may be updated by Dovetail from time to time, in print or
electronic form.

“Software” means the Co:Z Co-Processing Toolkit for z/OS, comprised of Co:Z
Launcher, Co:Z Dataset Pipes, Co:Z SFTP, Co:Z Batch, Co:Z ssh-proxyc
and Co:Z Target System Toolkit in object code form only, together with certain
sample code and scripts in source form.

“Update” means any bug fix, enhancement, or other modification to or update
for the Software issued by Dovetail for general release to the Software
community.

“You” means the person or entity downloading, installing or using the
Software.  If you are downloading, installing or using the Software on behalf
of a company or organization, the term “You” refers to both you and your
company or organization, and you represent and warrant that you have authority
to bind your company or organization to the provisions hereof.

2.  SOFTWARE LICENSE.  During the term of this Agreement, and subject to the
provisions hereof, Dovetail hereby grants to You, and You hereby accept, an
enterprise-wide, non-exclusive, non-transferable, royalty-free and fully
paid-up license to install and use the Software on an unlimited number of Your
servers, solely for Your internal business purposes, in accordance with the
Documentation, and in compliance with all applicable laws and regulations.

3.  LICENSE RESTRICTIONS.  You shall not, nor shall You authorize any other
person or entity to: (a) distribute, rent, lease, lend, sell, sublicense or
otherwise make the Software available to any third party; (b) modify, adapt,
alter, translate, or create derivative works of the Software; (c) use the
Software in or as part of a service bureau, timesharing or outsourcing
capacity; (d) develop an alternative to the Software that is based on or
derived from, in whole or in part, the Software or Documentation; (e) remove or
obscure any copyright, trademark or other proprietary rights notices or
designations on the Software, the Documentation or any copies thereof; or (f)
reverse engineer, decompile, disassemble, or otherwise attempt to derive the
```

source code for the Software, except where such reverse engineering is expressly permitted under applicable law, but then only to the extent that Dovetail is not entitled to limit such rights by contract.

4. UPDATES. From time to time, Dovetail may make available Updates for the Software as a general release to the Software community. All such Updates (whether posted by Dovetail on the Dovetail website or included with the Software) shall be deemed part of the Software, and are licensed to You under the license and other provisions of this Agreement, together with any supplementary license terms that Dovetail may provide for such Updates.

5. YOUR RESPONSIBILITIES. You are responsible for: (i) installation of the Software and any Updates; (ii) selecting and maintaining all third party hardware, software, peripherals and connectivity necessary to meet the system requirements for the Software; (iii) creating a restore point for Your systems and backing up and verifying all data; and (iv) adopting reasonable measures to ensure the safety, security, accuracy and integrity of Your facilities, systems, networks and data. Dovetail shall have no responsibility or liability arising out of or resulting in whole or in part from Your failure or delay to perform any such responsibilities, or for acts or omissions of third parties, Internet or telecommunications failures, or force majeure or other events beyond Dovetail’s reasonable control.

6. SUPPORT. This Agreement does not include, and Dovetail shall have no obligation under this Agreement to provide, any technical support or other professional services for the Software. If You are interested in purchasing a support plan for the Software, You should visit the Dovetail website to review Dovetail’s then current offerings.

7. TERM; TERMINATION. This Agreement and Your license rights hereunder shall continue unless and until terminated as set forth herein. You may terminate this Agreement for convenience at any time by uninstalling, erasing all copies of, and ceasing all use of the Software and Documentation. This Agreement shall terminate immediately and automatically if You violate the license terms or restrictions for the Software, or materially breach any other provision of this Agreement and fail to cure such breach within ten (10) days after receiving notice thereof from Dovetail. Upon the expiration or termination of this Agreement for any reason: (i) Your license to the Software shall automatically and immediately terminate; and (ii) You shall discontinue use of the Software, promptly (within 5 days) uninstall and remove any remnants of the Software and Documentation from Your computers, network and systems, and destroy (or return to Dovetail) all tangible copies of the Software and Documentation in Your possession. Sections 1, 3, 5, 7, 8, 9, 10 and 11 of this Agreement shall survive the expiration or termination of this Agreement for any reason, and shall be binding on and inure to the benefit of the parties and their permitted successors and assigns.

8. DISCLAIMER. THE SOFTWARE AND DOCUMENTATION ARE PROVIDED TO YOU UNDER THIS AGREEMENT “AS IS” WITHOUT REPRESENTATIONS OR WARRANTIES OF ANY KIND, EXPRESS OR IMPLIED, AND ALL USE IS AT YOUR OWN RISK. WITHOUT LIMITING THE FOREGOING, DOVETAIL AND ITS SUPPLIERS HEREBY disclaim any IMPLIED OR STATUTORY warranties of MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE OR NON-INFRINGEMENT. THE SOFTWARE IS NOT INTENDED OR LICENSED FOR USE IN ANY HAZARDOUS OR HIGH RISK ACTIVITY. DOVETAIL DOES NOT WARRANT THAT THE SOFTWARE WILL OPERATE UNINTERRUPTED OR ERROR-FREE, OR MEET YOUR BUSINESS, TECHNICAL OR OTHER REQUIREMENTS. No employee or agent has authority to bind DOVETAIL to any representations or warranties NOT EXPRESSLY SET FORTH IN THIS AGREEMENT.

9. PROPRIETARY RIGHTS. Dovetail and its suppliers shall retain exclusive

```
right, title and interest in and to the Software, including the object code,
source code, program architecture, design, coding methodology, Documentation,
screen shots, and “look and feel” therefor, all Updates thereto, all
goodwill associated therewith, and all present and future copyrights,
trademarks, trade secrets, patent rights and other intellectual property rights
of any nature throughout the world embodied therein and appurtenant thereto.
All rights and licenses to the Software not expressly granted to You in this
Agreement are reserved by Dovetail and its suppliers.  From time to time, You
may submit suggestions, requests or other feedback for the Software.  Dovetail
shall be free to commercialize and use such feedback, including for developing
improvements to its products and services, free of any claims, payment
obligations, or proprietary, confidentiality or other restrictions of any kind.


10.  LIMITATIONS ON LIABILITY.  IN NO EVENT SHALL DOVETAIL BE LIABLE FOR ANY
INDIRECT, INCIDENTAL, CONSEQUENTIAL, EXEMPLARY, SPECIAL, PUNITIVE OR SIMILAR
DAMAGES ARISING OUT OF OR RELATED TO THE SOFTWARE OR THIS AGREEMENT, INCLUDING
LOSS OF BUSINESS, PROFITS OR REVENUE, LOSS OR DESTRUCTION OF DATA, BUSINESS
INTERRUPTION OR DOWNTIME.  THE TOTAL CUMULATIVE LIABILITY OF DOVETAIL ARISING
OUT OF AND RELATED TO THE SOFTWARE AND THIS AGREEMENT SHALL NOT, REGARDLESS OF
THE NUMBER OF INCIDENTS OR CAUSES GIVING RISE TO ANY SUCH LIABILITY, EXCEED TEN
U.S. DOLLARS ($10).  THE LIMITATIONS ON LIABILITY IN THIS SECTION SHALL APPLY
TO THE MAXIMUM EXTENT PERMITTED UNDER APPLICABLE LAW, REGARDLESS OF THE CAUSE
OF ACTION OR BASIS OF LIABILITY (WHETHER IN CONTRACT, TORT OR OTHERWISE), AND
EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.  THESE LIMITATIONS ON
LIABILITY ARE AN ESSENTIAL PART OF THIS AGREEMENT, AND SHALL BE VALID AND
BINDING EVEN IF ANY REMEDY IS DEEMED TO FAIL OF ITS ESSENTIAL PURPOSE.

11.  MISCELLANEOUS

Governing Law.  This Agreement shall be governed and interpreted for all
purposes by the laws of the State of Missouri, U.S.A., without reference to any
conflict of laws principles that would require the application of the laws of a
different jurisdiction.  The United Nations Convention on Contracts for the
International Sale of Goods and the Uniform Computer Information Transactions
Act (as enacted in any jurisdiction) do not and shall not apply to this
Agreement, and are hereby specifically excluded.

Jurisdiction; Venue.  Any dispute, action or proceeding arising out of or
related to the Software or this Agreement shall be commenced in the state
courts of St. Louis County, Missouri or, where proper subject matter
jurisdiction exists, the United States District Court for the Eastern District
of Missouri.  Each party irrevocably submits and waives any objections to the
exclusive personal jurisdiction and venue of such courts, including any
objection based on forum non conveniens.

Notices.  All notices under this Agreement shall be in writing, and shall be
delivered personally or by postage prepaid certified mail or express courier
service, return receipt requested.  Notices to You may be delivered to the most
current address on file.  Notices to Dovetail shall be directed to the
following address, unless Dovetail has provided an alternative notice address:

Dovetailed Technologies, LLC
305 Willowpointe Drive
St. Charles, MO  63304

Assignments.  You may not assign or transfer this Agreement, or any rights or
duties hereunder, in whole or in part, whether by operation of law or
otherwise, without the prior written consent of Dovetail.  Any attempted
```

```
assignment or transfer in violation of the foregoing shall be null and void
from the beginning and without effect.  Dovetail may freely assign or transfer
this Agreement, including to a successor in interest upon Dovetail’s merger,
acquisition, corporate reorganization, or sale or other transfer of all or
substantially all of its business or assets to which this Agreement relates.

Relationship; Third Party Beneficiaries.  The parties hereto are independent
contractors.  Nothing in this Agreement shall be deemed to create any agency,
employment, partnership, fiduciary or joint venture relationship between the
parties, or to give any third party any rights or remedies under or by reason
of this Agreement; provided, however, the disclaimers and limitations on
liability in this Agreement shall extend to Dovetail and its directors,
officers, shareholders, employees, agents and affiliates.  All references to
Dovetail in connection therewith shall be deemed to include the foregoing
persons and entities, who shall be third party beneficiaries of such
contractual disclaimers and limitations and entitled to accept all benefits
afforded thereby.

Equitable Relief.  The Software comprises the confidential and proprietary
information of Dovetail and its suppliers, and constitutes a valuable trade
secret.  You acknowledge that Your breach of the license or ownership
provisions of this Agreement would cause irreparable harm to Dovetail, the
extent of which would be difficult and impracticable to assess, and that money
damages would not be an adequate remedy for such breach.  Accordingly, in
addition to all other remedies available at law or in equity, and as an express
exception to the jurisdiction and venue requirements of this Agreement,
Dovetail shall be entitled to seek injunctive or other equitable relief in any
court of competent jurisdiction.

U.S. Government Restricted Rights. The Software and Documentation are licensed
with RESTRICTED RIGHTS as "Commercial Items," as that term is defined at 48
C.F.R. §2.101, consisting of "Commercial Computer Software" and "Commercial
Computer Software Documentation," as such terms are used in 48 C.F.R. §12.212
or 48 C.F.R. §227.7202, as applicable. Consistent with 48 C.F.R. §12.212 or
48 C.F.R. §227.7202-1 through 227.7202-4, as applicable, the Commercial
Computer Software and Commercial Computer Software Documentation is licensed
(if at all) to U.S. Government end users only as Commercial Items, and with
only those rights as are granted to other licensees pursuant to this Agreement.

Export Control.  The Software and underlying information and technology may not
be accessed or used except as authorized by United States and other applicable
law, and further subject to compliance with this Agreement.  The Software may
not be exported or re-exported into any U.S. embargoed countries, or to anyone
on the U.S. Treasury Department's list of Specially Designated Nationals or the
U.S. Department of Commerce Denied Person’s List or Entity List.  You
represent and warrant that You and Your end users are not located in, under the
control of, or a national or resident of any country or on any such list.

Amendment; Waiver.  This Agreement may be amended only by a written instrument
signed by an authorized representative of Dovetail.  No rights shall be waived
by any act, omission or knowledge of a party, except by an instrument in
writing expressly waiving such rights and signed by an authorized
representative of the waiving party.  Any waiver on one occasion shall not
constitute a waiver on subsequent occasions.

Severability; Construction.  If any provision of this Agreement is determined
to be invalid or unenforceable under applicable law, such provision shall be
amended by a court of competent jurisdiction to accomplish the objectives of
such provision to the greatest extent possible, or severed from this Agreement
```

```
if such amendment is not possible, and the remaining provisions of this
Agreement shall continue in full force and effect.  The captions and section
headings in this Agreement are for reference purposes only and shall not affect
the meaning or interpretation of this Agreement.  The term “including” as
used herein means “including without limitation.”  The terms “herein,”
“hereto,” “hereof,” and similar variations refer to this Agreement as a
whole, rather than to any particular section.

Entire Agreement.  This Agreement sets forth the entire agreement of the
parties and supersedes all prior agreements and understandings, whether written
or oral, with regard to the subject matter hereof.  Any additional or
conflicting terms proposed by You in any purchase order, request for proposal,
acknowledgement, or other writing shall not be binding, and are hereby objected
to and expressly rejected.
```

Appendix E. References

E.1 z/OS OpenSSH

Using remote todsn and fromdsn clients requires that *z/OS OpenSSH* or *IBM Ported Tools OpenSSH* be available and configured on z/OS. z/OS V2R2 includes OpenSSH. Earlier versions of z/OS require IBM Ported Tools OpenSSH v1.2 (or later) to be installed. See the version of our *Quick Install Guides* matching your z/OS OpenSSH version for additional information.

E.2 Using the z/OS Unix Shell

The Dataset Pipes **todsn** and **fromdsn** commands may be used from any of the following z/OS Unix shell environments:

- The TSO "OMVS" command
- The **BPXBATCH** utility, running under MVS batch or TSO

  *Note:*The BPXBATCH enhancement **OA11699** significantly improves its usability.
- The z/OS Unix Shell under a telnet or ssh console.

For more information on z/OS Unix, see:

- *z/OS Unix System Services home*
- *z/OS Unix System Services User's Guide*

E.3 The z/OS C library fopen() routine

The Dataset Pipes utilties open MVS datasets in "record mode" using the z/OS C library fopen() routine. For example:

```
fopen( name, mode);
```

where:

name
: either `"//'fully.qualified.dsn'"` or `"//dd:ddname"` depending on whether **BPXWDYN** allocation keywords were used (*Section E.4, "The z/OS BPXWDYN dynamic allocation service"*).

mode

- `"rb,type=record,noseek"` - if reading (fromdsn)
- `"wb,type=record,noseek"` - if writing (todsn)
- `"ab,type=record,noseek"` - if appending (todsn -a)

Additional open mode options may be specifed by using the `-o` option.

The Dataset Pipes utilites read and write records using the z/OS C library `fread()` and `fwrite()` routines. For more information on the capabilities of record-mode dataset processing with the z/OS C library, see:

- *IBM z/OS C++ home*
- *z/OS V1R12 XL C/C++ Run-Time Library Reference*
- *z/OS V1R12 XL C/C++ Programming Guide*. See Ch. 10 "Performing OS I/O operations.

E.4 The z/OS BPXWDYN dynamic allocation service

The Dataset Pipes utilities allow for flexible allocation of MVS Datasets through use of the **BPXWDYN** text-based allocation service. If you specify allocation keywords, either with the `-x` option, or by using the `allocKeywords` option, then a new system-assigned DDNAME will be allocated with BPXWDYN and that DDNAME will be opened with *Section E.3, "The z/OS C library fopen() routine"*fopen().

You may use any allocation keywords defined by BPXWDYN, except the following:

- `DA(), DSN(), FI(), DD(), MSG(), or REUSE()` (automatically supplied)
- `PATH() , PATHDISP(), PATHMODE(), PATHOPTS(), PATHPERM()`
- `RTDDN, RTDSN, RTVOL` (only works if called from REXX)
- `SYNTAX`

For more information on using BPXWDYN allocation keywords, see:

- *z/OS V1R12 Using REXX and z/OS UNIX System Services*

E.5 The z/OS Unicode Translation Services

The Dataset Pipes utilities rely on the *z/OS Unicode Conversion Service* when possible, for codepage/character set translation.

This subsystem provides hardware-assisted high-performance codepage conversions services. This is the same service used by later versions of z/OS DB2, so many shops already have it configured in their environments. For z/OS 1.6 and later, the service is configured by default, with a starter set of codepage (CCSID) mappings.

For more information on configuring and customizing this subsystem:

- *z/OS V1R12 Unicode Services User's Guide and Reference*

When Unicode Conversion Services are not available, Dataset Pipes falls back to **iconv** for codepage translation