`binary`: Fixed and Variable Length Record Stream Access Method

CML00006-01

Code Magus Limited (England reg. no. 4024745)
Number 6, 69 Woodstock Road
Oxford, OX2 6EY, United Kingdom
`www.codemagus.com`

August 16, 2016

Contents

1 Introduction **2**

2 Open String Specification **2**
2.1 Open Specification Access Method Name 2
2.2 Open Specification Object Name 3
2.3 Open Specification Option Name-Value Pairs 3
2.4 Open String Specification Examples 3

3 `binary` access method definition file **3**

4 Environment **5**

1 Introduction

The `binary` access method is a module which implements the `recio` [1] provider interface allowing the `recio` user interface to support fixed and variable length record files implemented on byte-stream file systems. This includes VOS stream files, UNIX binary files, Windows/DOS binary files and z/OS classic MVS and HFS files. Classic MVS files can be supported either in their underlying record mode or by treating the files as binary files.

Two types of file are made available to calling systems. For fixed record length files the data in the underlying stream file system is written or read by the calling program one fixed length record at a time. The length of this record is determined by the `reclen` option value supplied in the open string specification supplied on the `recio_open` call.

For variable length records, the length of the record is supplied when the I/O operation takes place. Write operations are expected to supply the length of the data actually written and and a read operation (as with all `recio_read` function calls) returns the length of the actual record read.

The choice between variable and fixed length records is determined by the caller by supplying an appropriate value for the `recfm` option. Specify `recfm=f` for fixed length records (in which case a value of `lrecl` should also be supplied); and specify `recfm=v` for variable length records.

On byte-stream based file system files, variable length record files are implemented by the inclusion of a record descriptor word in front of each record. The format of this RDW is the same as the format on Classic MVS `RECFM=V` files. And is the same format that is included when these files are binary FTPed from MVS systems using the `quote site rdw` option. The actual records are read and written using the `rdwio` library.

2 Open String Specification

As with all `recio` library open specification strings, three components comprise the open string: access method, object, and options name-value pairs.

For the `binary` access method the access method name is `binary`.

2.1 Open Specification Access Method Name

The access method name should be specified as `binary`.

2.2 Open Specification Object Name

The object name should be a suitable file stream name on the local system. This is a stream file name appropriate to the local systems `fopen` file name parameter.

2.3 Open Specification Option Name-Value Pairs

Consult the access method definition file for the option name-value pairs supported by the `binary` access method. The access method definition file also supplies details of the default values (if any) of the options.

2.4 Open String Specification Examples

The following open string specification could be used to open a file as a variable length record file using the underlying file systems binary byte stream file. This string is suitable for opening the file for sequential input:

```
binary(/tmp/in,recfm=v,mode=rb)
```

The following open string specification could be used to open a file as a variable length record file using the underlying file systems binary byte stream file. This string is suitable for opening the file for sequential output:

```
binary(/tmp/out,recfm=v,mode=wb)
```

The following open string specification could be used to open a file as a fixed length record file using the underlying file systems binary byte stream file. This string is suitable for opening the file for sequential input:

```
binary(/tmp/out,recfm=f,mode=wb,lrecl=80)
```

3 `binary` access method definition file

The access method definition file should be consulted for the description of the options and their default values. This includes the description of the options. The access method definition file should also be consulted for the processing modes supported by the access method.

Refer to the `recio` library documentation for interpreting the contents of the access method definition file.

```
access binary(mode,recfm,reclen=32767,type="stream");

  -- File: BINARY.amd
```

```
--
-- This file contains an access method definition which is used to read
-- and write local files of fixed format records or rdw I/O library type
-- records.
--
-- Author: Stephen R. Donaldson [stephen@codemagus.com].
--
-- Copyright (c) 2008 Code Magus Limited. All rights reserved.

-- $Author: hayward $
-- $Date: 2012/08/02 11:14:31 $
-- $Id: BINARY.amd,v 1.11 2012/08/02 11:14:31 hayward Exp $
-- $Name:  $
-- $Revision: 1.11 $
-- $State: Exp $
--
-- $Log: BINARY.amd,v $
-- Revision 1.11  2012/08/02 11:14:31  hayward
-- Allow more MVS type open parameters.
--
-- Revision 1.10  2008/09/30 17:15:38  hayward
-- Allow append for output files.
--
-- Revision 1.9  2008/05/14 16:08:21  hayward
-- Split userdw and fixedlen into separate flags.
-- Fix typoss causing bugs.
--
-- Revision 1.8  2008/05/12 16:23:29  hayward
-- Add type= option. This will indicate that on record based OS's
-- rdwio should or should not be used.
--
-- Revision 1.7  2008/03/31 22:29:01  stephen
-- Add usage of environment variables to AMD file
--
-- Revision 1.6  2008/03/27 20:50:35  stephen
-- Windows updates
--
-- Revision 1.5  2008/03/27 20:46:33  stephen
-- Correct option name and option processing
--
-- Revision 1.4  2008/02/22 14:08:59  stephen
-- Correct name in access method definition
--
-- Revision 1.3  2008/02/22 13:41:54  stephen
-- Correct standard regular expressoin constraint
--
-- Revision 1.2  2008/02/22 11:40:47  stephen
-- Update for docuemenation
--

 modes seq_input, seq_output, skip_input;
```

```
   implements open;
   implements close;
   implements read;
   implements write;
   implements point;
   implements tell;

   describe mode as
      "The mode is the open mode string which will be passed to the C Standard "
      "I/O Library.";

   describe recfm as
      "The record format parameter describes the format of the records "
      "expected to be in the file. The records are either fixed length "
      "records (in which case a record length must be provided), or they "
      "are variable in length and in which case the record length is "
      "ignored (if specified).";

   describe reclen as
      "For fixed format records a record length is required. This parameter "
      "supplies the record length in bytes.";

   describe type as
      "This parameter supplies information as to the underlying operating "
      "system file structure. The option stream is the default and should "
      "be used for stream based operating systems. The option record must "
      "be used for operating systems that use a record based file structure "
      "(for example MVS) and should be used in conjunction with the sub "
      "option type= in the mode option.";

   constrain mode as "^[rwa]b\(,recfm=[AFUV][ABMS]*\)\?\(,type=record\)\?$";
   constrain recfm as "^[fv]$";
   constrain reclen as "^[1-9][0-9]*$";
   constrain type as "^\(stream\)\|\(record\)$";


   path = ${CODEMAGUS_AMDLIBS} "%s";
   module = "binaryam" ${CODEMAGUS_AMDSUFDL};
   entry = binaryam_init;

end.
```

4 Environment

The location and format of the access method definition file is required to be specified by the environment variable `CODEMAGUS_AMDPATH`. This environment variable supplies a pattern to the full path of where access method definition (or `amd`) files are located. The format of the environment variable is that of a path with a %s appearing in the position in which the access method member name should appear. For example, on

MVS systems this might have the form:

```
CODEMAGUS_AMDPATH='DNCT00.SRDA1.AMDFILES(%s)'
```

On a Unix-based system, the value might be set in a shell profile file such as:

```
export CODEMAGUS_AMDPATH=$HOME/bin/%s.amd
```

On Windows systems, the value might be supplied from the environment variables and look something like:

```
C:\CodeMagus\bin\%s.amd
```

References

[1] recio: Record Stream I/O Library Version 1. CML Document CML00001-01, Code Magus Limited, July 2008. PDF.