



LegacyJ DDS Plug-in for Java and PERCobol

The contents of this manual may be revised without prior notice. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical for any purpose without the expressed written permission of LegacyJ Corporation.

©Copyright LegacyJ Corp. 2001, 2005
All Rights Reserved.

Second Edition: June, 2005

LegacyJ has made every effort to ensure that this manual is correct and accurate, but reserves the right to make changes without notice at its sole discretion at any time.

© Copyright LegacyJ Corporation , 2001, 2005. All Rights Reserved

Trademarks

- PERCobol and LegacyJ are trademarks of LegacyJ Corporation.
- AIX, OS/390, OS/2 , OS/400 and iSeries are trademarks of International Business Machines.
- IBM is the registered trademark of International Business Machines.
- Java is a trademark of Sun Microsystems
- UNIX is a registered trademark licensed exclusively to X/Open Company Limited.
- Windows NT , Windows 2000 and Windows XP are trademarks of Microsoft Corporation.
- Other company, product and service names may be trademarks of service marks of others.

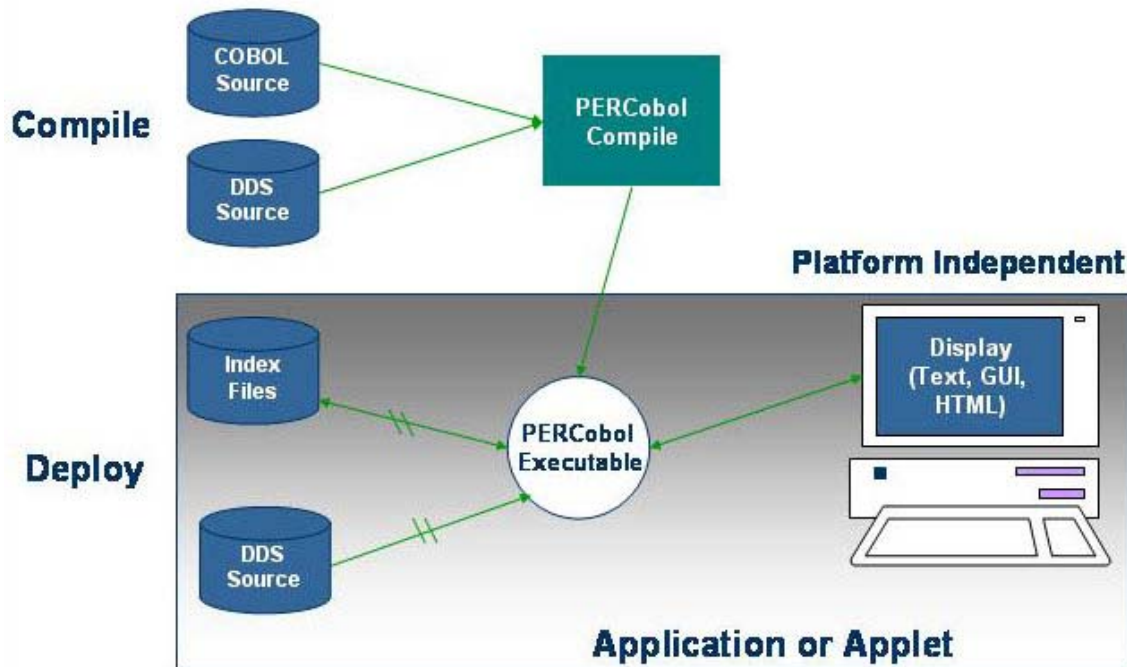
Contents

INTRODUCTION	1
ABOUT THE LEGACYJ DDS PLUG-IN	2
GETTING STARTED	3
JAVA	4
DEPLOY THE JAVA PROGRAM.....	5
SAMPLE PROGRAMS	6
PERCOBOL	7
RUNNING THE SAMPLE PROGRAM.....	8
APPENDIX A KEYWORDS SUPPORTED	9
APPENDIX B KEYWORDS NOT SUPPORTED	12
APPENDIX C DDS COLUMN DESCRIPTION	15

Introduction

The LegacyJ DDS Plug-in permits the control and display of OS/400 screen DDS (Data Description Specification) on any platform supporting Java. The Plug-in uses DDS source files to build screens. These screens can be deployed on the Web or as part of a platform independent client.

Using PERCobol with the Plug-in, screens are accessed with READ and WRITE statements just as with OS/400 COBOL. With Java, screens are controlled using a simple interface called the DDSController. The DDS plug-in supports a broad spectrum of screen DDS keywords, a detailed list is available in Appendix A. The unsupported keywords are listed in Appendix B.



PERCobol DDS Overview

About the LegacyJ DDS Plug-in

The LegacyJ DDS Plug-in uses display DDS source files for building screens. Although the plug-in does not use the compiled object files for building screens, source files should be compiled using the OS/400 CRTDSPF command and checked for possible errors prior to usage.

Source files are accessed as resources from the Java classpath. Therefore, these files should be available from the classpath and should be in the default text file encoding (ASCII / EBCDIC) for the platform.

The DDS Plug-in accesses message files such as QSYS/QCPFMSG to obtain error message information using one of two methods:

1. Using IBM's JTOpen toolbox to access message files residing on OS/400. For more information about the toolbox, visit:

<http://oss.software.ibm.com/developerworks/opensource/jt400/>

2. Using a simple text file accessible as a resource via the classpath:
 - a. The text file should be named:

```
<library name>_<message file name>, or just  
<message file name> if no library is specified.
```

The one exception to this rule is the QSYS/QCPFMSG file, which is accessed by the name 'QCPFMSG' rather than 'QSYS_QCPFMSG'.

- b. The text file should contain messages in this form:

```
<message id 1>=<message 1>  
<message id 2>=<message 2>  
...  
<message id n>=<message n>
```

Messages may contain substitution identifiers such as &1, &2, etc., to designate text substitution.

For example, a QCPFMSG file might contain lines such as:

```
CPD406A=Field value is not an allowed date. Reason code &1.  
CPD406B=Field value is not an allowed time. Reason code &1.  
CPF0197=Value &1 is not a valid name.  
KBD0007=Mandatory data entry field. Must have data entered.
```

The LegacyJ DDS Plug-in display screen is a character mode 5250-like display, emulated in Java. Since the display is a Java graphics component, it can be embedded in an Applet on a Web page. In PERCObol, this is automatically available when running the PERCObol program as an Applet.

Getting Started

All platforms

The LegacyJ DDS Plug-in is included in the LegacyJ IDE and must be activated prior to working with DDS files. If you do not have a registered key and would like to evaluate the product for 10 days, use the LegacyJ License Manager within the IDE to activate a temporary license.

Sample DDS programs are included in the LegacyJ product package demonstrate both COBOL and Java usage of the DDS Plug-in facility.

Java

A simple controller class, `com.legacyj.dds.DDSController`, is used to control display DDS from Java. This class should be imported into your Java program using the import statement:

```
import com.legacyj.dds.DDSController;
```

From within your program, instantiate the `DDSController` class using the name of the DDS file to access:

```
// Substitute MYFILE with actual DDS filename
DDSController myController=new DDSController("MYFILE");
```

Next, open the file for access using the open statement:

```
boolean result=myController.open();
```

The open statement returns true if successful. If the command returns false, check to make sure that the DDS source file is accessible from the classpath. Also check to make sure that the `legacyj.dat` license file is accessible from the user's home directory or the classpath.

Next, set the current record to be accessed:

```
myController.setCurrentRecord("MYRECORD");
```

In this example, we set the current record to a record called "MYRECORD". To retrieve a list of all the records in the file, use the `getRecordNames` method:

```
String[] recordNames=myController.getRecordNames();
```

Next, set the field and indicator values for the record:

```
myController.setFieldValue("MYFIELD1","ABC");
myController.setFieldValue("MYFIELD2","123");

myController.setIndicatorValue(1, true);
myController.setIndicatorValue(99, true);
```

In this case, "MYRECORD" has two output-capable fields, "MYFIELD1" and "MYFIELD2", which we set to "ABC" and "123" respectively. We also set option indicators 1 and 99 to true.

After setting field and indicator values, we perform a write-read operation:

```
int status;

status=myController.write();
if (myController.isError(status)) return false;

status=myController.read();
if (myController.isError(status)) return false;
```

In this example, both the write and read operations return integer status results. To check to see if the status is an error, we call the `isError` method.

Next we get the results of the read:

```
String value=myController.getFieldValueAsString("MYFIELD3");

boolean ind3=myController.getIndicatorValue(3);
boolean ind98=myController.getIndicatorValue(98);
```

In this example, we get the results from an input-capable field called "MYFIELD3", and two response indicators, 3 and 98.

Last, we close the file:

```
myController.close();
```

If problems occur, sometimes it is helpful to log trace information. The `DDSController` has a method called `setTraceMode` which can be used to enable or disable the output of trace information.

More information about these and other `DDSController` methods can be obtained from the javadocs under the `<ddsplugin home>/docs/javadoc` directory. Load `index.html` into your web browser to view the javadocs.

Deploy the Java program

The program is to be deployed using the LegacyJ Deploy Tool (Export Facility). This tool is included in the LegacyJ IDE and will package all the components required to run the DDS screens.

Sample programs

Two sample programs are provided for Java, DDSFile and DDSInteractive. The DDSFile sample demonstrates the different capabilities of the LegacyJ DDS Plug-in by allowing the user to run various DDS keyword tests from a menu. The DDSInteractive sample demonstrates many of the DDSController methods interactively. This sample also allows the user to test DDS screens to see how they will behave using the Plug-in.

PERCobol

PERCobol functions with the LegacyJ DDS Plug-in in the same manner that the OS/400 ILE COBOL handles DDS for display, using OPEN, CLOSE, WRITE, READ, WRITE SUBFILE, READ SUBFILE, READ NEXT SUBFILE, and REWRITE SUBFILE transaction file statements. Detailed information about ILE COBOL programming as well as DDS can be found online at:

<http://publib.boulder.ibm.com/pubs/html/as400/online/homeeng1.htm>

The first step in programming DDS in COBOL is to declare the DDS display file in the FILE-CONTROL section:

```
FILE-CONTROL.  
  SELECT TERMINAL-FILE  
    ASSIGN TO WORKSTATION-DDSFILE  
    ORGANIZATION IS TRANSACTION  
    ACCESS IS DYNAMIC  
    FILE STATUS IS F-STATUS  
    RELATIVE KEY IS SUBFILE-RECORD-NUMBER.
```

DDSFILE is the name of your DDS file in uppercase. The variables F-STATUS and SUBFILE-RECORD-NUMBER are values in working storage:

```
WORKING-STORAGE SECTION.  
.  
.  
01 F-STATUS PIC 99 VALUE 0.  
01 SUBFILE-RECORD-NUMBER PIC 99 VALUE 0.  
.  
.
```

Next, in the file section, declare the file descriptor and copy in the DDS records:

```
DATA DIVISION.  
FILE SECTION.  
FD TERMINAL-FILE  
  LABEL RECORDS ARE OMITTED.  
  01 TERMINAL-REC.  
  COPY DDS-ALL-FORMATS OF DDSFILE.
```

DDSFILE is the name of your DDS file given in the SELECT clause. This particular COPY statement commands PERCobol to generate COBOL data records from all record formats in the DDS file and COPY them into the COBOL program under TERMINAL-REC. PERCobol generates a text file with the extension ".ddscopy" in the working directory that contains the COBOL record values generated from the DDS file.

In the procedure division, open the file to access to the DDS file:

```
OPEN I-O TERMINAL-FILE
IF F-STATUS NOT EQUAL 0 THEN
    GO TO ERROR-EXIT
END-IF
```

Next, set the output-capable field and option indicator values for the record:

```
MOVE TRUE-VAL TO IN01 OF MYRECORD-O

MOVE "ABC" TO FIELD1 OF MYRECORD-O
MOVE "123" TO FIELD2 OF MYRECORD-O
```

In this example, a record in the DDS file named "MYRECORD" is represented by the COBOL record MYRECORD-O, produced using the COPY DDS-ALL-FORMATS phrase (see above).

To perform a write-read operation on the record "MYRECORD":

```
WRITE TERMINAL-REC FORMAT "MYRECORD"
IF F-STATUS NOT EQUAL 0 THEN
    GO TO ERROR-EXIT
END-IF

READ TERMINAL-FILE FORMAT "MYRECORD"
IF F-STATUS NOT EQUAL 0 THEN
    GO TO ERROR-EXIT
END-IF
```

Next, get the results of the read:

```
MOVE FIELD3 OF MYRECORD-I TO MYFIELDVALUE.
MOVE IN99 OF MYRECORD-I TO MYINDICATORVALUE.
```

In this case, "MYRECORD" has an input-capable field called "FIELD3" and a response indicator 99.

After all is complete, close the terminal file:

```
CLOSE TERMINAL-FILE
```

If problems occur, make certain that the ddsplugin.jar file is accessible from the classpath, and that the legacyj.dat license file is either in the user's home directory or accessible from the classpath.

To work properly, the DDS source file being accessed should compile without error using the CRTDSPF command on OS/400, and the file should use only keywords supported by the plug-in (see Appendix A).

Running the sample program

A sample programs is provided for PERCOBOL called DDSCOBOL.CBL. This sample demonstrates the different capabilities of the LegacyJ DDS Plug-in by allowing the user to run various DDS keyword tests from a menu.

Appendix A Keywords Supported

Keywords that are supported* in this version of the Plug-in:

*Keywords may be partially or fully supported.

Check Keywords

Keyword	Notes
CHANGE CHECK(AB) CHECK(ME) CHECK(MF) CHECK(Mx) CHECK(VN) CHECK(VNE) COMP RANGE VALUES CHGINPDMT	

Constant Keywords

Keyword	Notes
DATE DFT MSGCON SYSNAME TIME USER	

Control Keywords

Keyword	Notes
INVITE	

Display Keywords

Keyword	Notes
ALARM BLINK CLRL COLOR CSRLOC DATFMT DATSEP DSPATR(MDT) DSPATR(other) DSPATR(PR) DSPMOD DSPSIZ ENTFLDATR ERASE	*NOCURSORS not supported

Keyword	Notes
FLDCSRPRG FRCDTA INZRCD MAPVAL MSGALARM OVERLAY SLNO TIMFMT TIMSEP VALNUM	Supported in PERCobol but not Java.

Edit Keywords

Keyword	Notes
EDTCDE EDTWRD	

Help Keywords

Keyword	Notes
HLPARA HLPBDY HLPCLR HLPRCD HLPRTN	

Keyboard Keywords

Keyword	Notes
ALTHELP	
ALTPAGEDWN	
ALTPAGEUP	
Cann	
CFnn	
CHECK(ER)	
CHECK(FE)	
CHECK(LC)	
CHECK(RB)	
CHECK(RZ)	
CLEAR	Not supported on platforms that do not have a clear key
HELP	Use ALTHELP if your platform does not support help key
HLPCMDKEY	
HOME	Type ctrl-home
LOCK	
PAGEDOWN	
PAGEUP	
RETCMDKEY	
RETKEY	
RETLCKSTS	
VLDCMDKEY	

Misc. Keywords

Keyword	Notes
BLANKS DFTVAL INDARA FLTPCN RTNCSRLOC SETOF	

Message Keywords

Keyword	Notes
CHKMSGID ERRMSG ERRMSGID MSGID MSGLOC	

Subfile Control Keywords

Keyword	Notes
SFLCLR SFLCSRRRN SFLCTL SFLDLT SFLDSP SFLDSPCTL SFLINZ SFLPAG SFLRCDNBR SFLRNA SFLSIZ	

Subfile Record Keywords

Keyword	Notes
SFL SFLNXTCHG	

Appendix B Keywords not supported

Display keywords

Keyword	Notes
ALWGPH	
BLKFOLD	
CHECK(RL)	
CHECK(RLTB)	
CSRINPONLY	
DSPATR(OID)	
DSPATR(SP)	
DSPRL	
GRDATR	
GRDBOX	
GRDCLR	
GRDLIN	
GRDRCD	
MOUBTN	
USRDFN	
USRDSPMGT	
WRDWRAP	

Advanced Display keywords

Keyword	Notes
CHCACCEL	
CHCAVAIL	
CHCCTL	
CHCSLT	
CHCUNAVAIL	
CHOICE	
CNTFLD	
MLTCHCFLD	
MNUBAR	
MNUBARHC	
MNUBARDSP	
MNUBARSEP	
MNUBARSW	
MNUCNL	
PSHBTNCHC	
PSHBTNFLD	
PULLDOWN	
RMVWDW	
SNGCHCFLD	
WDWBORDER	
WDWTITLE	

Control keywords

Keyword	Notes
PASSRCD RTNDTA USRRSTDSP	

Edit Keywords

Keyword	Notes
EDTMSK FLTFIXDEC	

Help Keywords

Keyword	Notes
HLPDOC HLPEXCLD HLPFULL HLPPNLGRP HLPSCHIDX HLPSEQ HLPSELF HLPITITLE HLPID	

Keyboard keywords

Keyword	Notes
DUP PRINT UNLOCK	

Misc. Keywords

Keyword	Notes
ASSUME CCSID CHRID HTML KEEP LOGINP LOGOUT NOCCSID OPENPRT WINDOW	

Overlay keywords

Keyword	Notes
ALWROL ERASEINP INZINP MDTOFF	

Keyword	Notes
OVRATR OVRDTA PROTECT PUTOVR PUTRETAIN	

Reference Keywords

Keyword	Notes
ALIAS ALTNAME DLTCHK DLTEDT REF REFFLD	

Subfile message keywords

Keyword	Notes
ERRSFL	

Subfile keywords

Keyword	Notes
SFLCHCCTL SFLCSRPRG SFLMSGKEY SFLMSGRCD SFLPGMQ	

Subfile Control keywords

Keyword	Notes
SFLDROP SFLEND SFLENTER SFLFOLD SFLIN SFLMLTCHC SFLMODE SFLMSG SFLMSGID SFLROLVAL SFLRTNSEL SFLSCROLL SFLSNGCHC	

Text Keywords

Keyword	Notes
INDXT	

Appendix C DDS Column Description

POSITION	DESCRIPTION	
7	Asterisk = Comment	
7-16	Conditioning	
17	Type of Name or Specification R - Record, H - Help, OR Blank for Field Name	
18	Reserved	
19-28	Name	
29	Reference Function (R), Must be Field Level	
30-34	Length (Right Justify, Can Use +/- for Reference Field)	
35	Datatype/Keyboard Shift	
36-37	Keyboard Shift	Meaning
	blank	Default (DETERMINE DATATYPE)
	X	Alphabetic Only [A-Z,-] (CHAR)
	A	Alphanumeric Shift (CHAR)
	N	Numeric Shift (BOTH)
	S	Signed Numeric (NUMERIC)
	Y	Numeric Only [0-9+-. ,] (NUMERIC)
	W	Katakana (JAPAN ONLY) (CHAR)
	I	Inhibit Keyboard Entry (BOTH)
	D	Digits Only [0-9] (BOTH)
	M	Numeric Only CHAR [0-9+-. ,] (CHAR)
	F	Floating Point (NUMERIC)
	L	Date
	T	Time
	Z	Timestamp
36-37	Decimal Positions	(# of Digits Right of Decimal)
	blank	Character Datatype
	<NUMBER>	Zoned Decimal Datatype (Can use +/- for Ref Field)
38	Usage	
	Blank or O	Output Only
	I	Input Only
	B	Both Input and Output
	H	Hidden (Special Output and Input))
	M	Message (Special Output)
	P	Program-to-System (Special Output)
39-44	Location	
	Line = Positions 39-41	
	Column = Positions 42-44	
45-80	Keyword Entries	