

Oracle® Documaker

Implementing PDF417 Bar Codes

version 11.4

Part number: E14902-01

October 2009

Copyright © 2009, Oracle and/or its affiliates. All rights reserved.

The Programs (which include both the software and documentation) contain proprietary information; they are provided under a license agreement containing restrictions on use and disclosure and are also protected by copyright, patent, and other intellectual and industrial property laws. Reverse engineering, disassembly, or decompilation of the Programs, except to the extent required to obtain interoperability with other independently created software or as specified by law, is prohibited.

The information contained in this document is subject to change without notice. If you find any problems in the documentation, please report them to us in writing. This document is not warranted to be error-free. Except as may be expressly permitted in your license agreement for these Programs, no part of these Programs may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose.

If the Programs are delivered to the United States Government or anyone licensing or using the Programs on behalf of the United States Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS

Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the Programs, including documentation and technical data, shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement, and, to the extent applicable, the additional rights set forth in FAR 52.227-19, Commercial Computer Software--Restricted Rights (June 1987). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

The Programs are not intended for use in any nuclear, aviation, mass transit, medical, or other inherently dangerous applications. It shall be the licensee's responsibility to take all appropriate fail-safe, backup, redundancy and other measures to ensure the safe use of such applications if the Programs are used for such purposes, and we disclaim liability for any damages caused by such use of the Programs.

The Programs may provide links to Web sites and access to content, products, and services from third parties. Oracle is not responsible for the availability of, or any content provided on, third-party Web sites. You bear all risks associated with the use of such content. If you choose to purchase any products or services from a third party, the relationship is directly between you and the third party. Oracle is not responsible for: (a) the quality of third-party products or services; or (b) fulfilling any of the terms of the agreement with the third party, including delivery of products or services and warranty obligations related to purchased products or services. Oracle is not responsible for any loss or damage of any sort that you may incur from dealing with any third party.

Oracle, JD Edwards, and PeopleSoft are registered trademarks of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

THIRD PARTY SOFTWARE NOTICES

This product includes software developed by Apache Software Foundation (<http://www.apache.org/>).

THIS SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright © 2000-2009 The Apache Software Foundation. All rights reserved.

This product includes software distributed via the Berkeley Software Distribution (BSD) and licensed for binary distribution under the Generic BSD license.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright © 2009, Berkeley Software Distribution (BSD)

This product includes software developed by the JDOM Project (<http://www.jdom.org/>).

THIS SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE JDOM AUTHORS OR THE PROJECT CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright (C) 2000-2004 Jason Hunter & Brett McLaughlin. All rights reserved.

This product includes software developed by the Massachusetts Institute of Technology (MIT).

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Copyright © 2009 MIT

This product includes software developed by Jean-loup Gailly and Mark Adler. This software is provided 'as-is', without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of this software.

Copyright (c) 1995-2005 Jean-loup Gailly and Mark Adler

This software is based in part on the work of the Independent JPEG Group (<http://www.ijg.org/>).

This product includes software developed by the Dojo Foundation (<http://dojotoolkit.org>).

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright (c) 2005-2009, The Dojo Foundation. All rights reserved.

This product includes software developed by W3C.

Copyright © 2009 World Wide Web Consortium, (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University). All Rights Reserved. (<http://www.w3.org/Consortium/Legal/>)

This product includes software developed by Mathew R. Miller (<http://www.bluecreststudios.com>).

Copyright (c) 1999-2002 ComputerSmarts. All rights reserved.

This product includes software developed by Shaun Wilde and distributed via Code Project Open License (<http://www.codeproject.com>).

THIS WORK IS PROVIDED "AS IS", "WHERE IS" AND "AS AVAILABLE", WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES OR CONDITIONS OR GUARANTEES. YOU, THE USER, ASSUME ALL RISK IN ITS USE, INCLUDING COPYRIGHT INFRINGEMENT, PATENT INFRINGEMENT, SUITABILITY, ETC. AUTHOR EXPRESSLY DISCLAIMS ALL EXPRESS, IMPLIED OR STATUTORY WARRANTIES OR CONDITIONS, INCLUDING WITHOUT LIMITATION, WARRANTIES OR CONDITIONS OF MERCHANTABILITY, MERCHANTABLE QUALITY OR FITNESS FOR A PARTICULAR PURPOSE, OR ANY WARRANTY OF TITLE OR NON-INFRINGEMENT, OR THAT THE WORK (OR ANY PORTION THEREOF) IS CORRECT, USEFUL, BUG-FREE OR FREE OF VIRUSES. YOU MUST PASS THIS DISCLAIMER ON WHENEVER YOU DISTRIBUTE THE WORK OR DERIVATIVE WORKS.

This product includes software developed by Chris Maunder and distributed via Code Project Open License (<http://www.codeproject.com>).

THIS WORK IS PROVIDED "AS IS", "WHERE IS" AND "AS AVAILABLE", WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES OR CONDITIONS OR GUARANTEES. YOU, THE USER, ASSUME ALL RISK IN ITS USE, INCLUDING COPYRIGHT INFRINGEMENT, PATENT INFRINGEMENT, SUITABILITY, ETC. AUTHOR EXPRESSLY DISCLAIMS ALL EXPRESS, IMPLIED OR STATUTORY WARRANTIES OR CONDITIONS, INCLUDING WITHOUT LIMITATION, WARRANTIES OR CONDITIONS OF MERCHANTABILITY, MERCHANTABLE QUALITY OR FITNESS FOR A PARTICULAR PURPOSE, OR ANY WARRANTY OF TITLE OR NON-INFRINGEMENT, OR THAT THE WORK (OR ANY PORTION THEREOF) IS CORRECT, USEFUL, BUG-FREE OR FREE OF VIRUSES. YOU MUST PASS THIS DISCLAIMER ON WHENEVER YOU DISTRIBUTE THE WORK OR DERIVATIVE WORKS.

This product includes software developed by PJ Arends and distributed via Code Project Open License (<http://www.codeproject.com>).

THIS WORK IS PROVIDED "AS IS", "WHERE IS" AND "AS AVAILABLE", WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES OR CONDITIONS OR GUARANTEES. YOU, THE USER, ASSUME ALL RISK IN ITS USE, INCLUDING COPYRIGHT INFRINGEMENT, PATENT INFRINGEMENT, SUITABILITY, ETC. AUTHOR EXPRESSLY DISCLAIMS ALL EXPRESS, IMPLIED OR STATUTORY WARRANTIES OR CONDITIONS, INCLUDING WITHOUT LIMITATION, WARRANTIES OR CONDITIONS OF MERCHANTABILITY, MERCHANTABLE QUALITY OR FITNESS FOR A PARTICULAR PURPOSE, OR ANY WARRANTY OF TITLE OR NON-INFRINGEMENT, OR THAT THE WORK (OR ANY PORTION THEREOF) IS CORRECT, USEFUL, BUG-FREE OR FREE OF VIRUSES. YOU MUST PASS THIS DISCLAIMER ON WHENEVER YOU DISTRIBUTE THE WORK OR DERIVATIVE WORKS.

This product includes software developed by Erwin Tratar. This source code and all accompanying material is copyright (c) 1998-1999 Erwin Tratar. All rights reserved.

THIS SOFTWARE IS PROVIDED "AS IS" WITHOUT EXPRESS OR IMPLIED WARRANTY. USE IT AT YOUR OWN RISK! THE AUTHOR ACCEPTS NO LIABILITY FOR ANY DAMAGE/LOSS OF BUSINESS THAT THIS PRODUCT MAY CAUSE.

This product includes software developed by Sam Leffler of Silicon Graphics.

THE SOFTWARE IS PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EXPRESS, IMPLIED OR OTHERWISE, INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

IN NO EVENT SHALL SAM LEFFLER OR SILICON GRAPHICS BE LIABLE FOR ANY SPECIAL, INCIDENTAL, INDIRECT OR CONSEQUENTIAL DAMAGES OF ANY KIND, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER OR NOT ADVISED OF THE POSSIBILITY OF DAMAGE, AND ON ANY THEORY OF LIABILITY, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE

Copyright (c) 1988-1997 Sam Leffler
Copyright (c) 1991-1997 Silicon Graphics, Inc.

This product includes software developed by Guy Eric Schalnat, Andreas Dilger, Glenn Randers-Pehrson (current maintainer), and others. (<http://www.libpng.org>)

The PNG Reference Library is supplied "AS IS". The Contributing Authors and Group 42, Inc. disclaim all warranties, expressed or implied, including, without limitation, the warranties of merchantability and of fitness for any purpose. The Contributing Authors and Group 42, Inc. assume no liability for direct, indirect, incidental, special, exemplary, or consequential damages, which may result from the use of the PNG Reference Library, even if advised of the possibility of such damage.

This product includes software components distributed by the Cryptix Foundation.

THIS SOFTWARE IS PROVIDED BY THE CRYPTIX FOUNDATION LIMITED AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE CRYPTIX FOUNDATION LIMITED OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE

Copyright © 1995-2005 The Cryptix Foundation Limited. All rights reserved.

This product includes software components distributed by Sun Microsystems.

This software is provided "AS IS," without a warranty of any kind. ALLEXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE HEREBY EXCLUDED. SUN AND ITS LICENSORS SHALL NOT BE LIABLE FOR ANY DAMAGES SUFFERED BY LICENSEE AS A RESULT OF USING, MODIFYING OR DISTRIBUTING THE SOFTWARE OR ITS DERIVATIVES. IN NO EVENT WILL SUN OR ITS LICENSORS BE LIABLE FOR ANY LOST REVENUE, PROFIT OR DATA, OR FOR DIRECT, INDIRECT, SPECIAL, CONSEQUENTIAL, INCIDENTAL OR PUNITIVE DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF THE USE OF OR INABILITY TO USE SOFTWARE, EVEN IF SUN HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Copyright (c) 1998 Sun Microsystems, Inc. All Rights Reserved.

This product includes software components distributed by Dennis M. Sosnoski.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright © 2003-2007 Dennis M. Sosnoski. All Rights Reserved

It also includes materials licensed under Apache 1.1 and the following XPP3 license

THIS SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright © 2002 Extreme! Lab, Indiana University. All Rights Reserved

This product includes software components distributed by CodeProject. This software contains material that is © 1994-2005 The Ultimate Toolbox, all rights reserved.

This product includes software components distributed by Geir Landro.

Copyright © 2001-2003 Geir Landro (drop@destroydrop.com) JavaScript Tree - www.destroydrop.com/hjavadscripts/tree/version 0.96

This product includes software components distributed by the Hypersonic SQL Group.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE

Copyright © 1995-2000 by the Hypersonic SQL Group. All Rights Reserved

This product includes software components distributed by the International Business Machines Corporation and others.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Copyright (c) 1995-2009 International Business Machines Corporation and others. All rights reserved.

This product includes software components distributed by the University of Coimbra.

University of Coimbra distributes this software in the hope that it will be useful but DISCLAIMS ALL WARRANTIES WITH REGARD TO IT, including all implied warranties of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. In no event shall University of Coimbra be liable for any special, indirect or consequential damages (or any damages whatsoever) resulting from loss of use, data or profits, whether in an action of contract, negligence or other tortious action, arising out of or in connection with the use or performance of this software.

Copyright (c) 2000 University of Coimbra, Portugal. All Rights Reserved.

This product includes software components distributed by Steve Souza.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright © 2002, Steve Souza (admin@jamonapi.com). All Rights Reserved.

This product includes software developed by the OpenSymphony Group (<http://www.opensymphony.com/>.)"

Copyright © 2001-2004 The OpenSymphony Group. All Rights Reserved.

Contents

Chapter 1, Producing PDF417 Bar Codes with Documaker

- 3 Overview
- 5 Setting Up the PDF417 Solution
 - 5 Installing the Software on Windows
 - 7 Testing Your Installation on Windows
 - 7 Using CCCW32.DLL as a Replacement for CUSLIB (for versions 11.1 and lower)
 - 8 Installing the Software on MVS Systems
- 11 Implementing a PDF417 Solution
 - 11 Things You are Responsible For
 - 12 Implementation Team Responsibilities
 - 12 Oracle Product Development Responsibilities
 - 13 Ways to Implement the PDF417 Bar Code Solution
- 15 PDF417 Rules
 - 15 CreateNYAAMVADData
 - 18 INI Options
 - 21 CreatePDF417Barcode
 - 23 GetNYAAMVAVar
- 25 PDF417 IDS Rules
 - 25 P417NyPDF417
- 26 PDF417 DAL Functions
 - 27 P417DalCreateNYAAMVADData
 - 28 P417DalCreatePDF417Barcode
- 29 Input Data Structure
- 32 Output Data Structure
- 34 Examining the Output Data
- 38 Mapping Input Data to Output Data
- 40 Input Data Formatting Rules
- 41 PDF417 FXR Files
- 42 PDF417 Fonts

| | |
|----|---------------------------------|
| 44 | PDF417 Character Set |
| 45 | Sizing a PDF417 Bar Code |
| 48 | Determining the Columns per Row |
| 50 | Determining the Number of Rows |
| 52 | Tips |
| 52 | Producing Reliable Bar Codes |
| 53 | Turning on Tracing |
| 53 | Error Messages |

Chapter 1

Producing PDF417 Bar Codes with Documaker

This document describes how to create a PDF417 bar code that can contain any type of information. For instance, this capability makes your Documaker system compatible with the New York State Insurance Department's (NYSID) regulation that requires bar codes on driver ID cards.

NOTE: The ability to produce PDF417 bar codes is a custom add-on for Documaker versions 9.7 through 11.1, but was included in Documaker with the release of version 11.2. This document will note any differences.

Included is information on these topics:

- [Overview on page 3](#)
- [Setting Up the PDF417 Solution on page 5](#)
- [Implementing a PDF417 Solution on page 11](#)
- [PDF417 Rules on page 15](#)
- [PDF417 IDS Rules](#)
- [PDF417 DAL Functions on page 26](#)
- [Input Data Structure on page 29](#)
- [Output Data Structure on page 32](#)
- [Examining the Output Data on page 34](#)
- [Mapping Input Data to Output Data on page 38](#)
- [Input Data Formatting Rules on page 40](#)
- [PDF417 FXR Files on page 41](#)

- [PDF417 Fonts on page 42](#)
- [PDF417 Character Set on page 44](#)
- [Sizing a PDF417 Bar Code on page 45](#)
- [Determining the Columns per Row on page 48](#)
- [Determining the Number of Rows on page 50](#)
- [Tips on page 52](#)

OVERVIEW

In July, 2000, the state of New York announced the Insurance Information & Enforcement System (IIES) for the purpose of enforcing compulsory insurance laws.

A key element of the provision is that all insurance companies that provide auto insurance in the state of New York must add a bar code to driver ID cards and other documents.

The bar code must contain information that is specific to the insured and to the insurance company. The bar code is a public domain form of a 2-dimensional bar code known as *Portable Data File 417* or *PDF417*.

Oracle Insurance's PDF417 solution produces an industry standard PDF417 bar code which can be scanned by any industry standard scanner that supports PDF417 symbology. The Oracle applications along with Oracle-supplied fonts generate a valid and scannable bar codes you can print on AFP, Xerox, PostScript, and PCL printers.

Implementation timetable

Here is a summary of how this regulation was implemented, as reported in the document, *Encrypted 2D Bar Coded ID Card Update II*, which was published on September 29, 2000 by the New York State Department of Motor Vehicles.

| By this date | Milestone |
|---|---|
| August 28, 2000 | Begin certification of ID card samples generated from mainframe, non-mainframe and other vendor agency management systems. |
| October 6, 2000 (formerly September 2000) | Licensees (agents, brokers, agencies and insurance companies) notified by NYSID of procedures to obtain DMV software, an encryption key and PIN number. |
| November 13, 2000 (formerly November 1, 2000) | DMV systems in place to scan encrypted 2D bar coded ID cards. DMV will accept and scan/authenticate ID cards from licensees ready to issue. Only ID cards with fax compliant (large) bar codes can be faxed to customers, DMV, its agents, or partners for processing. If the bar code cannot be read after faxing, it is not acceptable. Faxed non-bar coded cards cannot be accepted. |
| March 5, 2001 | DMV received a number of requests to adjust the production deadlines via Draft Part 32 Regulation comments. Industry representatives confirmed the benefit of this final change at the recent IIES Advisory Committee meeting. DMV has adjusted the bar coded ID card roll-out timetable below. No additional delays will be provided. |
| October 1, 2001 (formerly scheduled for July 1, 2001.) | DMV, its agents and partners shall not accept an ID card for registration or insurance processing that does not contain a readable compliant encrypted 2D bar code. |
| January 1, 2002 (formerly October 1, 2001.) | Renewal ID cards issued with an effective date on or after January 1, 2002 must contain the compliant encrypted 2D bar code. All policyholders must have a compliant bar coded ID card by January 15, 2003 to show to law enforcement officers upon demand. |

If you have questions on the drivers ID cards, contact the New York State Department of Motor Vehicles at this email address:

2DIDcards@dmv.state.ny.us

Reference documents

These documents also provide information on this regulation:

| Title | Date | Published by |
|---|--------------------|---|
| <i>Guide to the Use of 2-D Bar Codes and Insurance ID Cards</i> | December 2000 | New York State Department of Motor Vehicles, Information Technology |
| <i>Programmer's Guide to use of IDCardGen Libraries</i> | January, 2001 | New York State Department of Motor Vehicles, Information Technology |
| <i>IIC Security Specifications</i> | June, 2000 | New York State Department of Motor Vehicles, Information Technology |
| <i>What's New — IDCardGen Module Changes</i> | January 16, 2000 | New York State Department of Motor Vehicles, Information Technology |
| <i>Encrypted 2D Bar Coded ID Card Update II</i> | September 29, 2000 | New York State Department of Motor Vehicles, Information Technology |
| <i>Best Practices Recommendation for the Use of Bar-Codes</i> | April 1996 | AAMVA - American Association of Motor Vehicle Administrators |

SETTING UP THE PDF417 SOLUTION

Getting the Non-disclosure Agreement

Oracle Insurance's PDF417 software, for use with Documaker Server, lets you generate the PDF417 bar code required by the New York State Insurance Department (NYSID) for automobile insurance ID cards.

You must promptly take the following actions for Oracle to prepare your custom software so you can produce PDF417 bar codes. Here are the steps you need to take:

1 Your company's compliance administrator must get and sign the non-disclosure agreement (NDA) from the NYSID, sign it, make a copy of the signed NDA, and return the original to the NYSID.

2 Send the signed copy of the NDA to Oracle Insurance at this address:

Oracle Insurance PDF417 Support
110 Abernathy Road
Building 500, Suite 1120
Atlanta, GA 30328

Building the solution

3 Oracle Insurance will then...

- Create PDF417 software customized to work in your environment with your Oracle applications.
- Send you the custom PDF417 software for use with your other Oracle applications.

If you have questions, contact Oracle Insurance Support at <http://metalink.oracle.com> or go to www.oracle.com/support/contact.html to find the appropriate phone number for your region.

NOTE: Beginning with version 11.2, this capability is incorporated into Documaker so there is no separate shipment or installation and you can create PDF417 bar codes without additional licensing. The implementation of the NYSID application, however, requires a security code that you can only get from the NYSID via the NDA process.

INSTALLING THE SOFTWARE ON WINDOWS

The PDF417 solution contains:

- Components and resources specific to a given platform
- Fonts
- Documentation
- Custom software

To install, copy the files on the CD into the appropriate directory on your computer. See the README.TXT file on the CD for detailed instructions.

NOTE: Version 11.2 includes the software components.

TESTING YOUR INSTALLATION ON WINDOWS

The PDF417 software includes a sample master resource library (MRL) you can use to make sure your installation works correctly. After you install the PDF417 software, follow these steps:

- 1 Go to the directory which contains the sample PDF417 master resource library. Typically, this will be the \PDF417_2 directory.
- 2 At an operating system prompt, enter one of these commands:

| To... | Enter... |
|--|----------------|
| See instructions | run |
| Test your system and create PCL output. | run pcl |
| Test your system and create PostScript output. | run pst |

You can see additional testing information on the README.TXT file in the PDF417_2 directory on the installation CD.

Using CCCW32.DLL as a Replacement for CUSLIB (for versions 11.1 and lower)

The CCCW32.DLL contains a copy of the base CUSLIB functionality plus the new PDF417 rules and functions. If you have not customized your system, you can substitute this file for CUSLIB as shown in the instructions below.

- 1 Install the base Documaker system.
- 2 Install the sample PDF417 master resource library.
- 3 Go to the directory where the Documaker DLL and EXE files are installed and rename the CUSW32.DLL file to *CUSW32.SAV*.
- 4 Copy the CCCW32.DLL file into same directory.
- 5 Rename the CCCW32.DLL file to *CUSW32.DLL*.
- 6 Test the results. See [Testing Your Installation on Windows on page 7](#) for more information.

When you finish testing, delete the CUSW32.DLL file and rename the CUSW32.SAV file to *CUSW32.DLL*.

NOTE: If you have customized your system, you must add the new functionality to your customized library.

Version 11.2 and higher includes the software as a part of the base product and the use of CCCLIB and CUSLIB has been discontinued.

INSTALLING THE SOFTWARE ON MVS SYSTEMS

NOTE: Version 11.2 and higher includes the PDF417 software and the use of CCCLIB and CUSLIB has been discontinued. The following information pertains to prior versions is included to help you migrate from prior version to version 11.2.

Before you install the software on your MVS system, make sure you...

- Are running the GenData program on an MVS or OS/390 operating system.
- Have IBM's C/C++ compiler for MVS/ESA, version 3.2 or above.
- Have a modified CUSLIB—you have written custom rules for your site and have added those rules to the CUSLIB portion of DAP.

Follow these instructions to install and test the PDF417 software on an MVS system:

- 1** Unload the contents of the MVS cartridge onto your MVS system.

You can find the JCL to unload the files on the tape in member LOADPDF, in the first file of the tape. Here is an example:

```
//JWCI JOB (33005), 'LOAD LOADPDF', CLASS=T, MSGCLASS=X,
//      NOTIFY=JWC
// * * * * *
// * LOAD - COPY THE LOADPDF JOB FROM TAPE TO MVS.
// *
// *
// * * * * *
//S1 EXEC PGM=IEBCOPY
//SYSPRINT DD SYSOUT=*
//INDD DD DSN=FSI.V102.JCLLIB, DISP=SHR, UNIT=CART,
//      VOL=SER=P10201
//OUTDD DD DSN=FSI.V102.JCLLIB.INST, DISP=(, CATLG), UNIT=SYSDA,
//      SPACE=(CYL, (1, 1, 1))
//SYSIN DD *
C I=INDD, O=OUTDD
SELECT MEMBER=LOADPDF
//*
```

- 2** Copy the PDF417 bar code-related CUSLIB modules into your CUSLIB and update the CUSREG member of your CUSLIB INCLIB.

Either copy the following files from FSI.V102.CCCLIB.SOURCE to your existing CUSLIB SOURCE PDS or create another PDF417 CUSLIB SOURCE PDS and copy these files into that CUSLIB SOURCE PDS:

```
CCCAAMVA
CCCP417
```

Either copy the following file from FSI.V102.CCCLIB.INCLIB into your existing CUSLIB INCLIB PDS or create another PDF417 CUSLIB INCLIB PDS and copy the file into that CUSLIB INCLIB PDS:

```
CCCAAMVA.H
```

The CUSREG member of your CUSLIB INCLIB PDS contains registration entries for custom rules shipped with the system as well as entries for any rules you have written and added to your CUSLIB.

You must add the new rule registration entries for PDF417 to the CUSREG member of your CUSLIB INCLIB PDS. You can find these entries at the bottom of the CUSREG member.

Copy the CreateNYAAMVADData and CreatePDF417Barcode entries into the ImageCustomRuleParms array in the CUSREG member of your CUSLIB INCLIB PDS:

```
ImageFuncSym aImageCustomRuleParms EMPTY() =

...

{"CreateNYAAMVADData",      CreateNYAAMVADData},
{"CreatePDF417Barcode",     CreatePDF417Barcode},
{"\0",                      NULL}
```

The last entry in the aImageCustomRuleParms array should remain as “\0”.

3 Recompile your CUSLIB.

Since the previous step modified the CUSREG header file in your INCLIB, you must recompile all of the modules in your CUSLIB, including the new CCCAAMVA and CCCP417 modules.

In step 2, if instead of copying the SOURCE and INCLIB files into your existing CUSLIB PDSs, you copied these files into the newly-created PDF417 SOURCE and INCLIB PDSs, compile the modules in your new PDF417 CUSLIB SOURCE PDS and in your existing CUSLIB SOURCE PDS.

NOTE: Make sure the CUSREG header file (in INCLIB) that you updated with the new registration entries (in step 2) is the one the compile job is using.

4 Link-edit a new GenData.

NOTE: Be sure to save your existing GenData executable (in LINKLIB) so you can use it in case you have problems with the new GenData.

Step 3 recompiled your CUSLIB, adding the new rules for PDF417 bar code processing. To make these rules available to GenData, link-edit a new GenData, making sure that the CUSLIB OBJECT modules created in step 3 are the ones used in the link-edit.

You can use the LINKCCC member in the JCLLIB PDF to perform the link-edit of GenData. Note that the LINKCCC job calls the LINKCCC proc, which references several additional libraries, such as the OBJLIBs for Symbol, NYAAMVA and CCCLIB and the SYS1.SCEECPP library for C++.

5 Run the PDF417 bar code installation verification test.

Some PDF417 sample resources are provided on the MVS cartridge. To run with these resources, run the DAPCCC job in JCLLIB. Be sure to edit the FSISYS INI file in DEFLIB to generate the appropriate type of print output, such as Xerox or AFP.

The DAPCCC job is similar to the normal DAP job except the location of the GenData executable is overridden. Replace the name of the STEPLIB referenced in the DAPCCC job with the name of the library where you link-edited the GenData executable in step 4.

IMPLEMENTING A PDF417 SOLUTION

There are numerous steps involved in implementing a PDF417 bar code solution for the NYSID application.

Things You are Responsible For

- Signing the NDA and sending a copy to Oracle Insurance and the NYSID.

NOTE: For implementations based on 11.2 or higher, you do not have to send a copy of the NDA to Oracle Insurance.

- Serving as the authorized contact (*a go-between*) for the implementation team to get answers to questions that can only be answered by the NYSID.
- Providing the implementation team with a list of forms requiring the addition of the bar code, or initiating a requirements study to make that determination.
- Providing the implementation team with access to the current MRL, including forms, rules, and data definitions for the production system so necessary changes can be made.
- Providing the implementation team with the source code to the current application software in production (such as any CUSLIB source code) into which the new custom rules can be added.

NOTE: For implementations based on 11.2 or higher, you do not have to integrate the software into CUSLIB.

- Authorizing your IT group (or other internal control group) to install the new PDF417 fonts provided by Oracle for the high-volume printers, network printers, workstations, or servers.
- Getting the company's unique security key from the NYSID and providing it to the implementation team.
- Providing the implementation team with the authorizations and access to local facilities to perform the necessary implementation and testing, including login rights, and access to test data regions.
- Receiving the print out test results from the implementation team and submitting them to the NYSID for approval as a part of the certification process.
- Receiving and relaying the results of the certification tests back to the implementation team, and re-submitting subsequent tests as necessary until certification is achieved.
- Managing the migration of the updated programs and resources into production.

Implementation Team Responsibilities

The Implementation Team consist of you or Oracle Professional Services.

- Determining which forms will need to have the bar code added and the size, location, and purpose of the bar code. (That is, choosing the font size based on business requirement.)
- Analyzing the existing forms to determine if you can add the bar code to them or if redesign work is necessary. Performing the redesign, if necessary.
- Analyzing the form data to determine if changes are necessary to make the data meet the bar code requirements.
- Analyzing the extract data to determine if the necessary data is available or if some re-implementation work is necessary.
- Integrating the two new rules into CUSLIB. You first integrate the rules on the PC and then port the code to MVS or UNIX as necessary.

NOTE: For implementations based on version 11.2 and higher, you do not have to integrate the code into CUSLIB.

- Modifying the scripts, DDT files, INI files, and so on, as necessary to invoke the new rules, in the batch system or the online system.

NOTE: DDT files may not be applicable to versions 11.0 and higher.

- Installing the printer or display fonts, or providing them to IT for installation, as necessary for either online or batch implementations.
- Testing the implementation, getting output that can be successfully scanned.
- Delivering clean test output for submission to the NYSID for certification.
- Making subsequent changes as needed until certification is achieved.
- Assisting as necessary in moving the implementation into production.

Oracle Product Development Responsibilities

- Providing the code required for the two processes: creating the New York bar code information and creating a PDF417 bar code. The exact code provided will vary based on the product and platform.

For the Documaker product line, code is provided for two CUSLIB rules for batch implementations and object libraries for the New York application and the PDF417 bar code technology.

Documaker Workstation and IDS implementations will receive DLL modules.

NOTE: Implementations based on version 11.2 or higher do not require customization.

- Providing the necessary fonts for either batch printing, printing from archive, or embedding in PDF, as needed.
- Providing documentation and other support to make sure the implementation is successful.
- Working with Oracle to make sure the necessary knowledge is available and transmitted to implementation team members.

WAYS TO IMPLEMENT THE PDF417 BAR CODE SOLUTION

There are several ways you can implement a PDF417 bar code solution in a Documaker environment. For instance, you can do it via...

- Documaker batch processing
- Archive retrieve and view
- Documaker Workstation
- IDS (iDocumaker Workstation, iPPS, or other DPR usage)

Here is a brief overview of the necessary steps

Batch processing

To implement a batch processing solution, you must...

- 1** Add two rules to CUSLIB.

NOTE: Version 11.2 and higher includes these rules in the base RULLIB. Do not place them in CUSLIB. If you are upgrading to 11.2, remove this code from CUSLIB.

- 2** Update your FAP, DDT, and INI files.
- 3** Use printer fonts (PCL, AFP, PostScript, or Metacode).

Archive retrieve and view

To implement an archive retrieve and view solution, you must...

- 1** Use PCL fonts for printing.
- 2** Use TTF fonts for viewing purposes (optional) or faxing (required).

Documaker Workstation

To implement a Documaker Workstation solution, you must...

- 1** Use PCL fonts for printing.
- 2** Use TTF fonts for faxing (required) or viewing (optional).
- 3** Add the new DLL modules into the environment. (You may have to re-compile the modules for older, unsupported releases, such as those prior to version 9.7.)
- 4** Implement calls to user callout procedures (DLL functions) in the FAP files.

IDS (iDocumaker
Workstation/iPPS)

- 1** Use TTF fonts for embedding in PDF.
- 2** Add the new DLL module to the server environment.
- 3** Update rule configurations to invoke the new IDS rule at all locations necessary to create the bar codes in the necessary data import implementation.
- 4** Test printing using Adobe Acrobat.

Since each scenario uses different software, platforms, and fonts, each may require certification by New York.

PDF417 RULES

The PDF417 software for Documaker includes these CUSLIB section level rules:

NOTE: These rules are included in version 11.2 (RULLIB).

- [CreateNYAAMVADData](#)
- [CreatePDF417Barcode on page 21](#)
- [GetNYAAMVAVar on page 23](#)

The first rule lets you create the data while the second rule lets you create the bar code. These processes are separate to enhance performance. The third rule lets you retrieve members of the AAMVA structure after it is built. See [Turning on Tracing on page 53](#) for debugging information.

NOTE: The CCCW32.DLL contains a copy of the base CUSLIB functionality plus the PDF417 rules and functions.

Beginning with version 11.2, CCCLIB and CUSLIB are no longer applicable.

CREATENYAAMVADATA

This rule is supported as a base CUSLIB rule. To receive this rule, you must:

- Be authorized to produce auto insurance policies in the state of New York.
- Have executed the non-disclosure agreement with New York.
- Have licensed the bar code technology from Oracle.

If you meet these criteria, Oracle will provide this rule for integration into CUSLIB, either by the your staff or by Oracle Professional Services.

NOTE: Beginning with version 11.2, these rules are included in the base product.

The CreateNYAAMVADData rule uses application and encryption routines contained in source code provided by the state of New York for purposes of porting or including into custom applications.

Oracle provides an API wrapper to this technology for general-purpose use within Oracle's publishing products, but Oracle does not warrant or expose all of the functionality contained in that software.

Syntax `;CreateNYAAMVAData; GVMOutputVariable , CheckForRequired , INS_Key;`

Parameters

| Parameter | Description |
|------------------------------------|--|
| GVMOutputVariable | <p>Specifies the name for the GVM variable that will contain the 570 bytes of formatted data be stored into a PDF417 bar code by a later call.</p> <p>You can make several calls using this data to create bar codes of different sizes and locations, but which contain the same data.</p> <p>Keep in mind that this rule is typically run once, while you can call the CreatePDF417Barcode rule several times using the same source GVM data.</p> |
| CheckForRequired | <p>The New York application defines a number of variable data elements to be included in the bar code. Some of these elements are required. This parameter lets you decide when and to what extent to check for these required data elements.</p> <p>If you enter one (1), if any required fields are missing from the data, an error is issued for each missing field.</p> <p>If you enter two (2), if any required fields are missing from the data, a warning is issued for each missing field.</p> <p>If you enter three (3), processing continues and no errors or warnings are issued if required fields are missing.</p> <p>The default is 3.</p> <p>Other INI options control whether the Rules Processor continues processing or stops on errors and warnings. These options are documented in the Documaker Server System Guide.</p> |
| INS_Key | <p>Each insurance company that writes auto insurance for drivers in New York will receive a unique key signature. This value is an <i>armored</i> hex string. This parameter specifies that key.</p> <p>If you omit this parameter, you must include this information in the INI file. For security reasons, you may want to have the source code to this rule customized to hardcode the constant key value into the rule. The base version does not provide this option.</p> <p>If you use the INI file to provide the information, specify the value as shown here:</p> <pre>< NYAAMVA> Ins_Key = value</pre> |
| PassPhrase (implicit parameter) | <p>The INS_KEY parameter is an armored 36-character hex string value. Internally, it must be de-armored using a <i>pass phrase</i> before it can be used. The rule does not support pass phrase as an explicit rule parameter, but the pass phrase is an implicit parameter necessary for compliance, and it must be specified in the INI file. The PassPhrase is a plain text message string that can only be obtained by an authorized insurance company from the NYSID. For security reasons, you may want to have the source code to the rule customized to hardcode the constant pass phrase string value into the rule. The base version does not provide this option.</p> <p>Specify the pass phrase value in the INI file as follows:</p> <pre>< NYAAMVA > PassPhase = value</pre> |

The rule begins by finding and destroying any existing GVM variable named by the `GVMOutputVariable` parameter.

- 1 The rule provides an interface between the Documaker batch process and application logic developed and provided by the NYSID. To interface with the New York application code, an interface structure must be filled out. This definition of this internal structure is specified by the New York application, and is called *InsuranceInfo* in C, and Oracle provides a COBOL version known as *NYID-INSURANCE-INFO*.

The rule automates the process of moving data within the Rules Processor into the proper position in the interface structure. The rule uses an internal table of variable names and has the necessary offsets and lengths to assemble the structure. The internal table also has flags to indicate which fields are required.

- 2 The rule checks each field in the table. The rule first looks in the INI file to see if you have specified alternate mapping. The rule uses the name as the option and looks in the INI file. If found, it examines the value to see if you specified cross-mapping. You can specify a variable field, a constant value, a GVM variable, a DAL variable, a DAL script, or any other existing built-in INI function. If there is no INI entry for the variable, the rule assumes the name will be used as is and it then looks for a field with that name.
- 3 The rule first looks for a variable field with a given name on the current section (FAP file). If the field is found on the section, that field will be used. If the field does not have any data or is blank, then the matching structure element will also be blank.

If the field is not found on the section, the rule next searches the entire document set for an occurrence of that named field which contains data. If no occurrence of the field can be found with data, the rule looks for a GVM variable with that name. If still no data is found, the rule looks for a DAL variable with that name. If it does not find a DAL variable, the search ends and no data will be used.

If an appropriate value is found, the rule uses the offset and length stored in the table to move the data into the interface structure. If no data was found, the corresponding structure member is left blank.

If the structure member is empty (blank), the rule examines the `CheckForRequired` parameter to see what action should be taken.

| If the <code>CheckForRequired</code> parameter is | Then |
|---|-------------------------------|
| 1 and the data is empty | An error is issued |
| 2 | A warning is issued |
| 3 | No warning or error is issued |

If an error is issued, other INI options for the Rules Processor determine whether processing continues or stops.

- 4 Once the *InsuranceInfo* structure has been assembled, the structure is passed in to the New York application API, along with the `INS_Key` parameter, the `PassPhrase` option, and the output structure, which is called *Insurance.AAMVA*.

- 5 The InsuranceAAMVA output structure contains 570 bytes of formatted data to be encoded into the bar code. The rule stores this formatted data in a GVM variable, named by the GVMOutputVariable parameter, for later use.

INI Options

The easiest implementation is one that provides the required data elements using the reserved names assigned by Oracle. These names are also used as C structure names in the New York application. Below, for convenience, a table provides a cross-reference of these names to COBOL structure member names used by other Oracle Insurance applications.

Many implementations will have some data elements available under different names. Some data elements require manipulation before being included into the bar code. To avoid requiring an existing form to be re-implemented or changes to the extract file, you may want to map the reserved variable names to other names or other sources of data.

This rule supports a number of INI options and ways to configure each of the 30 or so variables that are necessary to feed the New York structure to alternate sources.

```
< NYAAMVA >
  RLastName1      = field name
  RFirstName1     = ;constant value
  RMiddleName1   = ;
  RNameSuffix1   = ;
  RNameAsOne1    = ;
  RClientId1     = ;~(any existing INI built-in)
  RLastName2     =
  RFirstName2    =
  RMiddleName2   =
  RNameSuffix2   =
  RNameAsOne2    =
  RClientId2     =
  RStreet        =
  RCity          =
  RState         =
  RZipCode       =
  ROrganizationLine1=
  ROrganizationLine2=
  ROrganizationFein=
  VVinNumber     =
  VYear          =
  VMake          =
  VRepInd        =
  VHistInd       =
  VTowInd        =
  VSeats         =
  IIssuerId      =
  IInsCompanyCode=
  IInsIssuanceDate=
  ICoverageStartDate=
  ICoverageEndDate=
  IPolicyNumber  =
```

Using the INI mapping options

The INI file offers many ways to get the necessary interface variables. Here are some examples:

```
< NYAAMVA >
  RLastName1= LAST NAME
```

The above example looks for data named LAST NAME. The rule first looks for a field on this section. Then it looks for a global field. Next, it looks for a GVM variable. Finally, it looks for a DAL variable.

```
VTowInd      = ;N
```

The above example shows how a company that never insures tow trucks could short cut this part and hardcode a constant N as the tow truck indicator value.

This table shows all of the variables accessed via the table, the corresponding C-structure member name, and whether the variable is required.

| NYID-INSURANCE-INFO name Used by the COBOL copy book record structure. | Corresponding NY InsuranceInfo C-structure member name and INI option. | Required? |
|--|--|-----------|
| NYID-R-LAST-NAME-1 | RLastName1 | Yes |
| NYID-R-FIRST-NAME-1 | RFirstName1 | Yes |
| NYID-R-MIDDLE-NAME-1 | RMiddleName1 | No |
| NYID-R-NAME-SUFFIX-1 | RNameSuffix1 | No |
| NYID-R-NAME-AS-ONE-1 | RNameAsOne1 | Yes |
| NYID-R-CLIENT-ID-1 | RClientId1 | Yes |
| NYID-R-LAST-NAME-2 | RLastName2 | No |
| NYID-R-FIRST-NAME-2 | RFirstName2 | No |
| NYID-R-MIDDLE-NAME-2 | RMiddleName2 | No |
| NYID-R-NAME-SUFFIX-2 | RNameSuffix2 | No |
| NYID-R-NAME-AS-ONE-2 | RNameAsOne2 | No |
| NYID-R-CLIENT-ID-2 | RClientId2 | No |
| NYID-R-STREET | RStreet | Yes |
| NYID-R-CITY | RCity | Yes |
| NYID-R-STATE | RState | Yes |
| NYID-R-ZIP-CODE | RZipCode | Yes |
| NYID-R-ORGANIZATION-LINE-1 | ROrganizationLine1 | Yes |
| NYID-R-ORGANIZATION-LINE-2 | ROrganizationLine2 | No |

| NYID-INSURANCE-INFO name Used by the COBOL copy book record structure. | Corresponding NY InsuranceInfo C-structure member name and INI option. | Required? |
|--|--|-----------|
| NYID-R-ORGANIZATION-FEIN | ROrganizationFEIN | Yes |
| NYID-V-VIN-NUMBER | VVinNumber | Yes |
| NYID-V-YEAR | VYear | Yes |
| NYID-V-MAKE | VMake | Yes |
| NYID-V-REP-IND | VRepInd | Yes |
| NYID-V-HIST-IND | VHistInd | No |
| NYID-V-TOW-IND | VTowInd | No |
| NYID-V-SEATS | VSeats | No |
| NYID-I-ISSUER-ID | IIssuerID | Yes |
| NYID-I-INS-COMPANY-CODE | IInsCompanyCode | Yes |
| NYID-I-INS-ISSUANCE-DATE | IInsIssuanceDate | Yes |
| NYID-I-COVERAGE-START- DATE | ICoverageStartDate | Yes |
| NYID-I-COVERAGE-END-DATE | ICoverageEndDate | Yes |
| NYID-I-POLICY-NUMBER | IPolicyNumber | Yes |

CREATEPDF417BARCODE

You can include this rule in CUSLIB for supported releases. Many customers have customized versions of CUSLIB, so the source code to the rules will be made available for integration into these implementations.

NOTE: The rule source is provided for DAP and Documaker versions 9.7 and higher. If you are using older releases, you may experience compatibility issues with these rules. If you are using one of these older releases, be aware that support for these systems is billable.

The CreatePDF417Barcode rule uses technology contained in a PDF417 bar code library acquired from Symbol Technology, the inventors of the PDF417 standard. This library is provided in object form only, and source code will not be distributed.

Oracle provides an API wrapper to this technology for general purpose use within Oracle Insurance's publishing products, but Oracle does not warrant or expose all of the functionality contained in that software.

Syntax

```
;CreatePDF417Barcode;N=BarCodeName,F=FontID,E=ECCLevel,G=GVMSourceVariable,T=TopFAPCoord,L=LeftFAPCoord,B=BottomFAPCoord,R=RightFAPCoord,W=Warning;
```

| Parameter | Description |
|-------------|--|
| BarCodeName | Specifies a name for the bar code. This name can be the name of a box object located on the current section. It is required to provide the root name used by a series of dynamic text labels that contain the bar code text data. |
| FontID | Specifies the PDF417 font you want to use for this instance of the bar code. You can choose between two sizes of PDF417 bar codes fonts: 9 dots high by 12 dots wide and 12 dots high by 16 dots wide. While Oracle does not mandate the IDs you can use, it does provide an FXR file which contains the attributes for the two fonts. The IDs for these fonts are: 912 and 1216. Note: Prior to release 11.2, the PDF417_2.FXR and PDF417.FXR files were provided to licensed PDF417 customers. In release 11.2, the new REL112.FXR file contains the PDF417 font references found in the two PDF417 FXR files previously included in this product. See for more information on the included FXR files, see PDF417 FXR Files on page 41 . |
| ECCLevel | Specifies the degree of error correction to be used in the bar code. The range is from zero (0) to eight (8). The default is four (4), which is the value recommended by the NYSID. For more information, see Sizing a PDF417 Bar Code on page 45 . |

| Parameter | Description |
|--|--|
| GVMSourceVariable | <p>Specifies the name of a GVM variable which contains the data to be embedded into the bar code. The CreateNYAAMVADData rule must have previously created this data and stored it in this GVM variable. The New York application currently specifies 570 bytes of data.</p> <p>It is likely that you will need to use the same source data to create a bar code multiple times on the same page, or on multiple forms in the form set. The bar codes can be different sizes and use different fonts yet contain the same source data. This is why it is important for performance that data creation and bar code creation steps are performed by separate rules.</p> |
| TopFAPCoord LeftFAPCoord BottomFAPCoord RightFAPCoord | <p>Each bar code instance must specify a location and a rectangular size. There are two ways to specify this information:</p> <ul style="list-style-type: none"> - from a named box object on the FAP file - from explicit rectangle FAP coordinates <p>The rectangle coordinate parameters are optional. When included, the coordinates explicitly provide the dimensions of the rectangular area to contain the bar code. If you omit the rectangle parameters, the system loads the FAP file for this section and searches for a box named BARCODENAME.</p> <p>For optimal performance, specify the rectangle's coordinates.</p> |
| Warning | <p>If the system cannot locate the specified bar code or box for the rule and the section is a multi-page section, you will get a warning. To suppress this warning, set this parameter to No (W=No). The default is Yes.</p> |

NOTE: Do not specify a location and a rectangular size using explicit rectangle coordinates in a multi-page section. This causes the system to create the bar code on every page of the multi-page section.

This rule performs these actions:

- 1 The rule first checks to see if the rectangle coordinates were provided. The most important ones specify the bottom and right. If either the bottom or right are zero or blank, the system assumes the coordinates were omitted and the FAP file must be loaded.

If the section has to be loaded, the rule tries to locate a box object in the section. The box must have the name specified in the BarCodeName parameter.
- 2 The rule uses the rectangle coordinates that were specified explicitly or specified by the named box. Using these coordinates, the rule determines the height and width of the bar code.
- 3 The rule uses the font specified in the FontID parameter to locate the requested bar code font. The FXR font information provides the cell height, width, and baseline values.
- 4 The rule uses the font cell width to calculate the number of text characters (COLS) that can fit within the width of the bar code rectangle.

- 5 The rule uses the cell height to calculate the number of text character text labels (ROWS) that can fit within the height of the bar code rectangle.
- 6 The rule checks for and destroys any existing bar code text. These text labels will be internally named with a root name matching the name specified by BarCodeName.
- 7 The rule retrieves the bar code data from a GVM variable specified by the GVMSourceVariable parameter.
- 8 The rule passes the bar code data, along with the calculated number of rows and columns, and the requested error correction level, to internal APIs that create the text characters necessary to render the bar code.

An error at this point could mean that the bar code area was too small for the source data and ECC level.
- 9 The rule creates dynamic text labels, row by row, for all of the returned bar code text. Each dynamic text label will have a root name specified by the BarCodeName parameter, the font specified by the FontID parameter, and its cell parameters, data length specified by COLS, and will contain the text for that row.

The dynamic text labels are unloaded automatically into the NAFILE.DAT file.

GETNYAAMVAVAR

This rule lets you retrieve members of the AAMVA structure after it is built. Certain fields in the AAMVA structure that are not supplied by the user are resolved during the processing of the CreateNYAAMVADData rule. This rule lets you retrieve the data from those fields and place it in a field or a GVM variable.

NOTE: The field must exist before this rule is executed. You can create a field using the MK_Hard rule on the field in the DDT file. You cannot create an empty field.

Syntax ;GetNYAAMVAVar;GVMOutputVariable,AAMVAMember, Field/
GVMVariable;

You can retrieve more than one AAMVA data member in a single call to GetNYAAMVAVar. To do this, simply use another pair of AAMVAMEMBER and FIELD/GVMVARIABLE combinations.

| Parameter | Description |
|-------------------|---|
| GVMOutputVariable | This is the name of the GVM variable supplied to CreateNYAAMVADData where the AAMVA structure is stored. |
| AAMVAMember | This is the name of the AAMVA field you want to retrieve. It should match the names used in the INI file, such as RNameAsOne1, RLastName2, and so on. |
| Field/GVMVariable | This is where the data will be placed. If you are sending the data to a field, must use the F switch, and for GVM variables use the V switch. |

Example ;GetNYAAMVAVar;G=CustomerInfo,RNameAsOne1,F=FULLNAMEFIELD,
 RLastName2,V=SECONDNAME;

In this example, the rule gets the *RNameAsOne1* data from the CustomerInfo AAMVA data and places it in the field *FULLNAMEFIELD*. It will also get the *RLastName2* data member and place it in the GVM variable *SECONDNAME*.

PDF417 IDS RULES

To implement a PDF417 bar code solution with IDS, you use the P417NyPDF417 rule. This rule handles both gathering the data and building the bar code. See [Turning on Tracing on page 53](#) for debugging information.

P417NyPDF417

Use this rule to create PDF417 bar codes in IDS implementations.

Syntax

P417NyPDF417 FontID, ECCLevel, BarCodeName

| Parameter | Description |
|-----------------------------|--|
| FontID | <p>Specifies PDF417 font you want to use for this instance of the bar code. You can choose between two sizes of PDF417 bar codes fonts: 9 dots high by 12 dots wide and 12 dots high by 16 dots wide.</p> <p>While Oracle does not mandate the IDs you can use, it does provide an FXR file which contains the attributes for the two fonts. The IDs for these fonts are: 912 and 1216.</p> <p>Note: Prior to release 11.2, the PDF417_2.FXR and PDF417.FXR files were provided to licensed PDF417 customers. In release 11.2, the new REL112.FXR file contains the PDF417 font references found in the two PDF417 FXR files previously included in this product.</p> <p>See for more information on the included FXR files, see PDF417 FXR Files on page 41.</p> |
| ECCLevel | <p>Specifies the degree of error correction to be used in the bar code. The range is from zero (0) to eight (8). The default is four (4), which is the value recommended by the NYSID.</p> <p>For more information, see Sizing a PDF417 Bar Code on page 45.</p> |
| BarCodeName (or BoxName) | <p>Specifies a name for the bar code or box. This name can be the name of a box object located on the current section. It is required to provide the root name used by a series of dynamic text labels that contain the bar code text data.</p> |

The rule works by first searching for all the boxes with the name given in the BarCodeName parameter. It then builds the bar code by looking at the INI file, then the current section, and then the form set.

Once it makes the bar code, the rule populates it for all boxes in the section. When the rule finds a box that is in a different section, it remakes the bar code. So you can make each bar code different by putting them in different sections.

Add this rule under the ReqType control group in the DOCSERV.INI file as shown here:

```
< ReqType: SAMPCO >
  function = dpros2->DPRLoadImportFile
  function = p417w32->P417NyPDF417, 1216, 4, barcode
  function = dpros2->DPRPrint
```

where SAMPCO is the group name. Be sure to add this rule after the DPRLoadImportFile rule.

PDF417 DAL FUNCTIONS

Registering the functions

You can also use the following DAL functions to implement a PDF417 solution:

- P417DalCreateNYAAMVAData
- P417DalCreatePDF417Barcode

Since these DAL functions are not registered by default, you must register them in the INI file. This will look something like this:

```
< DALFunctions >
  Keyword = DLL->FunctionName
```

Where *Keyword* left of the equals sign represents the DAL script function name you want to register.

To the right of the equals sign, *DLL->FunctionName* represents a DLL to load and the exported function name associated with the script function.

Registered in this manner, you can then reference these functions in a DAL script just like any other built-in function. For instance, you could register these functions as shown here:

```
< DALFunctions >
  CreateNYAAMVAData = P417W32->P417DalCreateNYAAMVAData
  CreatePDF417Barcode= P417W32->P417DalCreatePDF417Barcode
```

And then refer to them as `CreateNYAAMVAData()` and `CreatePDF417Barcode()` in your DAL scripts.

You could also register the functions as shown here:

```
< DALFunctions >
  NYAAMVA = P417W32->P417DalCreateNYAAMVAData
  PDF417 = P417W32->P417DalCreatePDF417Barcode
```

And then refer to them as `NYAAMVA()` and `PDF417()` in your scripts. How you register these functions is up to you, just make sure the name you use is descriptive.

Both DAL functions work virtually identical to the PDF417 rules.

P417DALCREATENYAAMVADATA

Use this function to create the data you want to place in the bar code.

Syntax

```
P417DalCreateNYAAMVADData (DALVariableName, RequiredLevel, INS_Key)
```

| Parameters | Description |
|-----------------|---|
| DALVariableName | <p>Specifies the name for the DAL variable that will contain the 570 bytes of formatted data be stored into a PDF417 bar code by a later call.</p> <p>You can make several calls using this data to create bar codes of different sizes and locations, but which contain the same data.</p> <p>Keep in mind that this function is typically run once, while the CreatePDF417Barcode function can be called several times using the same source DAL data.</p> |
| RequiredLevel | <p>The New York application defines a number of variable data elements to be included in the bar code. Some of these elements are required. This parameter lets you decide when and to what extent to check for these required data elements.</p> <p>If you enter one (1), if any required fields are missing from the data, an error is issued for each missing field.</p> <p>If you enter two (2), if any required fields are missing from the data, a warning is issued for each missing field.</p> <p>If you enter three (3), processing continues and no errors or warnings are issued if required fields are missing.</p> <p>The default is 3.</p> <p>Other INI options control whether the Rules Processor continues processing or stops on errors and warnings. These options are documented in the Rules Processor System Guide.</p> |
| INS_Key | <p>Each insurance company that writes auto insurance for drivers in New York will receive a unique key signature. This value is an <i>armored</i> hex string. This parameter specifies that key.</p> <p>If you omit this parameter, you must include this information in the INI file. For security reasons, you may want to have the source code to this rule customized to hardcode the constant key value into the rule. The base version does not provide this option.</p> <p>If you use the INI file to provide the information, specify the value as shown here:</p> <pre>< NYAAMVA> Ins_Key = value</pre> |

This function returns DALERR_SUCCESS if successful. Otherwise, it returns DALERR_USER and shows a dialog.

P417DALCREATEPDF417BARCODE

Syntax `P417DalCreatePDF417Barcode (DALVariableName, FontID, BarCodeName or BoxName, ECCLevel, TopCoord, BottomCoord, LeftCoord, RightCoord)`

| Parameter | Description |
|--|---|
| DALVariableName | <p>Specifies the name for the DAL variable that will contain the 570 bytes of formatted data be stored into a PDF417 bar code by a later call.</p> <p>You can make several calls using this data to create bar codes of different sizes and locations, but which contain the same data.</p> <p>Keep in mind that this function is typically run once, while the CreatePDF417Barcode function can be called several times using the same source DAL data.</p> |
| FontID | <p>Specifies PDF417 font you want to use for this instance of the bar code. You can choose between two sizes of PDF417 bar code fonts: 9 dots high by 12 dots wide and 12 dots high by 16 dots wide.</p> <p>While Oracle does not mandate the IDs you can use, it does provide an FXR file which contains the attributes for the two fonts. The IDs for these fonts are: 912 and 1216.</p> <p>Note: Prior to release 11.2, the PDF417_2.FXR and PDF417.FXR files were provided to licensed PDF417 customers. In release 11.2, the new REL112.FXR file contains the PDF417 font references found in the two PDF417 FXR files previously included in this product.</p> <p>See for more information on the included FXR files, see PDF417 FXR Files on page 41.</p> |
| BarCodeName (or BoxName) | <p>Specifies a name for the bar code or box. This name can be the name of a box object located on the current section. It is required to provide the root name used by a series of dynamic text labels that contain the bar code text data.</p> |
| ECCLevel | <p>Specifies the degree of error correction to be used in the bar code. The range is from zero (0) to eight (8). The default is four (4), which is the value recommended by the NYSID.</p> <p>For more information, see Sizing a PDF417 Bar Code on page 45.</p> |
| TopFAPCoord LeftFAPCoord BottomFAPCoord RightFAPCoord | <p>Each bar code instance must specify a location and a rectangular size. There are two ways to specify this information:</p> <ul style="list-style-type: none"> - from a named box object on the FAP file - from explicit rectangle coordinates <p>The rectangle coordinate parameters are optional. When included, the coordinates explicitly provide the dimensions of the rectangular area to contain the bar code. If you omit the rectangle parameters, the system loads the FAP file for this section and searches for a box named BarCodeName.</p> <p>For optimal performance, specify the rectangle's coordinates.</p> |

This function returns DALERR_SUCCESS if successful. Otherwise, it returns DALERR_USER and shows a dialog.

INPUT DATA STRUCTURE

The New York state application code expects to receive the NYID-INSURANCE-INFO input data structure. This data structure should be in a native format (such as EBCDIC on MVS, ASCII on Windows NT) and will be converted to ASCII, if necessary by Oracle routines.

NOTE: The New York state required data formatting is adopted from public industry standards. These formatting requirements are accomplished in the software you get from the NYSID. These formats include the American Association of Motor Vehicle Administrators (AAMVA) compliant bar code, MD5 message digest algorithms, and the public domain encryption algorithm called *Triple DES* which generates the required internal digital signature.

The New York application code is written in C/C++ and uses many conventions specific to those languages. Many Oracle implementations will use COBOL programs to interface to this application code. Therefore, part of the interface project involves inter-language communication. Oracle provides COBOL layouts that match the format expected by the C/C++ programs, and will work across the inter-language procedure calls.

The COBOL copybook record structure is defined in NYINSINF.CBL as shown below. COBOL application programs should first set the structure to all NULLs (low values such as binary zero).

This makes sure the FILLER members remain set to NULL as each data member is assigned. As each data member is assigned a value, COBOL will blank fill any unused character positions. The receiving program (written in C/C++) will use a structure view and treat each field as a null-terminated C string.

```

01  NYID-INSURANCE-INFO.
    02  NYID-REGISTRANT-SUB-FILE.
        04  NYID-R-LAST-NAME-1          PIC X(18) .
        04  FILLER                      PIC X.
        04  NYID-R-FIRST-NAME-1        PIC X(16) .
        04  FILLER                      PIC X.
        04  NYID-R-MIDDLE-NAME-1      PIC X(16) .
        04  FILLER                      PIC X.
        04  NYID-R-NAME-SUFFIX-1      PIC X(3) .
        04  FILLER                      PIC X.
        04  NYID-R-NAME-AS-ONE-1      PIC X(20) .
        04  FILLER                      PIC X.
        04  NYID-R-CLIENT-ID-1       PIC X(9) .
        04  FILLER                      PIC X.
        04  NYID-R-LAST-NAME-2       PIC X(18) .
        04  FILLER                      PIC X.
        04  NYID-R-FIRST-NAME-2      PIC X(16) .
        04  FILLER                      PIC X.
        04  NYID-R-MIDDLE-NAME-2     PIC X(16) .
        04  FILLER                      PIC X.
        04  NYID-R-NAME-SUFFIX-2     PIC X(3) .
        04  FILLER                      PIC X.
        04  NYID-R-NAME-AS-ONE-2     PIC X(20) .
        04  FILLER                      PIC X.
        04  NYID-R-CLIENT-ID-2      PIC X(9) .
        04  FILLER                      PIC X.

```

| | | |
|----|------------------------------|-------------|
| 04 | NYID-R-STREET | PIC X(20) . |
| 04 | FILLER | PIC X. |
| 04 | NYID-R-CITY | PIC X(15) . |
| 04 | FILLER | PIC X. |
| 04 | NYID-R-STATE | PIC X(2) . |
| 04 | FILLER | PIC X. |
| 04 | NYID-R-ZIP-CODE | PIC X(9) . |
| 04 | FILLER | PIC X. |
| 04 | NYID-R-ORGANIZATION-LINE-1 | PIC X(20) . |
| 04 | FILLER | PIC X. |
| 04 | NYID-R-ORGANIZATION-LINE-2 | PIC X(20) . |
| 04 | FILLER | PIC X. |
| 04 | NYID-R-ORGANIZATION-FEIN | PIC X(9) . |
| 04 | FILLER | PIC X. |
| 02 | NYID-VEHICLE-SUB-FILE . | |
| 04 | NYID-V-VIN-NUMBER | PIC X(25) . |
| 04 | FILLER | PIC X. |
| 04 | NYID-V-YEAR | PIC X(4) . |
| 04 | FILLER | PIC X. |
| 04 | NYID-V-MAKE | PIC X(5) . |
| 04 | FILLER | PIC X. |
| 04 | NYID-V-REP-IND | PIC X(1) . |
| 04 | FILLER | PIC X. |
| 04 | NYID-V-HIST-IND | PIC X(1) . |
| 04 | FILLER | PIC X. |
| 04 | NYID-V-TOW-IND | PIC X(1) . |
| 04 | FILLER | PIC X. |
| 04 | NYID-V-SEATS | PIC X(2) . |
| 04 | FILLER | PIC X. |
| 04 | NYID-V-VIN-OVERRIDE | PIC X(1) . |
| 04 | FILLER | PIC X. |
| 02 | NYID-INSURANCE-SUB-FILE . | |
| 04 | NYID-I-DOCUMENT-TYPE | PIC X(15) . |
| 04 | FILLER | PIC X. |
| 04 | NYID-I-PAPER-SELECTION | PIC X(15) . |
| 04 | FILLER | PIC X. |
| 04 | NYID-I-ISSUER-ID | PIC X(10) . |
| 04 | FILLER | PIC X. |
| 04 | NYID-I-INS-COMPANY-CODE | PIC X(3) . |
| 04 | FILLER | PIC X. |
| 04 | NYID-I-INS-COMPANY-NAME | PIC X(40) . |
| 04 | FILLER | PIC X. |
| 04 | NYID-I-INS-ISSUANCE-DATE | PIC X(8) . |
| 04 | FILLER | PIC X. |
| 04 | NYID-I-COVERAGE-START-DATE | PIC X(8) . |
| 04 | FILLER | PIC X. |
| 04 | NYID-I-COVERAGE-END-DATE | PIC X(8) . |
| 04 | FILLER | PIC X. |
| 04 | NYID-I-POLICY-NUMBER | PIC X(15) . |
| 04 | FILLER | PIC X. |
| 04 | NYID-I-AGENCY-NAME | PIC X(40) . |
| 04 | FILLER | PIC X. |
| 04 | NYID-I-AGENCY-ADDRESS-LINE-1 | PIC X(40) . |
| 04 | FILLER | PIC X. |
| 04 | NYID-I-AGENCY-ADDRESS-LINE-2 | PIC X(40) . |


```
04 FILLER PIC X.
04 NYID-I-IS-ORGANIZATION PIC X(1) .
04 FILLER PIC X.
02 NYID-DIGITAL-CERT-SUB-FILE.
04 NYID-S-SIGNATURE-TYPE PIC X(4) .
04 FILLER PIC X.
04 NYID-S-DIGITAL-SIGNATURE PIC X(32) .
04 FILLER PIC X.
```

OUTPUT DATA STRUCTURE

The New York state ID application code creates the NY-AAMVA-INSURANCE-INFO output data structure. This data structure, in ASCII format, represents the actual content of the PDF417 bar code that will be printed on the ID card.

The COBOL copybook record structure is defined in NYAAMVA.CBL as follows:

```

01 NY-AAMVA-INSURANCE-INFO.
   02 NY-AAMVA-HEADER-DEF.
      04 NY-AAMVA-CMPL-IND          PIC X(1) .
      04 NY-AAMVA-DATA-SEP          PIC X(1) .
      04 NY-AAMVA-RECORD-SEP        PIC X(1) .
      04 NY-AAMVA-SEGMENT-TERM      PIC X(1) .
      04 NY-AAMVA-FILE-TYPE         PIC X(5) .
      04 NY-AAMVA-IIN               PIC X(6) .
      04 NY-AAMVA-VERSION-NUMBER    PIC X(2) .
      04 NY-AAMVA-NO-OF-SUBFILES    PIC X(2) .
   02 NY-AAMVA-SUBFILE-DESIGNATOR.
      04 NY-AAMVA-SUB-REG-TYPE       PIC X(2) .
      04 NY-AAMVA-REG-OFFSET        PIC X(4) .
      04 NY-AAMVA-REG-LEN           PIC X(4) .
      04 NY-AAMVA-SUB-VEH-TYPE      PIC X(2) .
      04 NY-AAMVA-VEH-OFFSET        PIC X(4) .
      04 NY-AAMVA-VEH-LEN           PIC X(4) .
      04 NY-AAMVA-SUB-INS-TYPE      PIC X(2) .
      04 NY-AAMVA-INS-OFFSET        PIC X(4) .
      04 NY-AAMVA-INS-LEN           PIC X(4) .
      04 NY-AAMVA-SUB-SIG-TYPE      PIC X(2) .
      04 NY-AAMVA-SIG-OFFSET        PIC X(4) .
      04 NY-AAMVA-SIG-LEN           PIC X(4) .
   02 NY-AAMVA-REGISTRANT-SUBFILE.
      04 NY-AAMVA-SUB-REG-START      PIC X(2) .
      04 NY-AAMVA-DES-R-LAST-NAME-1 PIC X(4) .
      04 NY-AAMVA-SUB-R-LAST-NAME-1 PIC X(18) .
      04 NY-AAMVA-DES-R-GIVEN-NAME-1 PIC X(4) .
      04 NY-AAMVA-SUB-R-GIVEN-NAME-1 PIC X(16) .
      04 NY-AAMVA-DES-R-MIDDLE-INIT-1 PIC X(4) .
      04 NY-AAMVA-SUB-R-MIDDLE-INIT-1 PIC X(1) .
      04 NY-AAMVA-DES-R-NAME-SUFX-1  PIC X(4) .
      04 NY-AAMVA-SUB-R-NAME-SUFX-1  PIC X(3) .
      04 NY-AAMVA-DES-R-NAME-AS-ONE-1 PIC X(4) .
      04 NY-AAMVA-SUB-R-NAME-AS-ONE-1 PIC X(20) .
      04 NY-AAMVA-DES-R-CLIENT-ID-1  PIC X(4) .
      04 NY-AAMVA-SUB-R-CLIENT-ID-1  PIC X(9) .
      04 NY-AAMVA-DES-R-LAST-NAME-2  PIC X(4) .
      04 NY-AAMVA-SUB-R-LAST-NAME-2  PIC X(18) .
      04 NY-AAMVA-DES-R-GIVEN-NAME-2 PIC X(4) .
      04 NY-AAMVA-SUB-R-GIVEN-NAME-2 PIC X(16) .
      04 NY-AAMVA-DES-R-MIDDLE-INIT-2 PIC X(4) .
      04 NY-AAMVA-SUB-R-MIDDLE-INIT-2 PIC X(1) .
      04 NY-AAMVA-DES-R-NAME-SUFX-2  PIC X(4) .
      04 NY-AAMVA-SUB-R-NAME-SUFX-2  PIC X(3) .
      04 NY-AAMVA-DES-R-NAME-AS-ONE-2 PIC X(4) .
      04 NY-AAMVA-SUB-R-NAME-AS-ONE-2 PIC X(20) .
      04 NY-AAMVA-DES-R-CLIENT-ID-2  PIC X(4) .
      04 NY-AAMVA-SUB-R-CLIENT-ID-2  PIC X(9) .
      04 NY-AAMVA-DES-R-ORG-LINE-1   PIC X(4) .

```

```

04 NY-AAMVA-SUB-R-ORG-LINE-1 PIC X(20) .
04 NY-AAMVA-DES-R-ORG-LINE-2 PIC X(4) .
04 NY-AAMVA-SUB-R-ORG-LINE-2 PIC X(20) .
04 NY-AAMVA-DES-R-ORG-FEIN PIC X(4) .
04 NY-AAMVA-SUB-R-ORG-FEIN PIC X(9) .
04 NY-AAMVA-DES-R-SEATS PIC X(4) .
04 NY-AAMVA-SUB-R-SEATS PIC X(2) .
04 NY-AAMVA-DES-R-STREET PIC X(4) .
04 NY-AAMVA-SUB-R-STREET PIC X(20) .
04 NY-AAMVA-DES-R-CITY PIC X(4) .
04 NY-AAMVA-SUB-R-CITY PIC X(15) .
04 NY-AAMVA-DES-R-STATE PIC X(4) .
04 NY-AAMVA-SUB-R-STATE PIC X(2) .
04 NY-AAMVA-DES-R-ZIP-CODE PIC X(4) .
04 NY-AAMVA-SUB-R-ZIP-CODE PIC X(5) .
04 NY-AAMVA-DES-REG-END PIC X(1) .
02 NY-AAMVA-VEHICLE-SUBFILE .
04 NY-AAMVA-SUB-VEH-START PIC X(2) .
04 NY-AAMVA-DES-V-VIN-NUMBER PIC X(4) .
04 NY-AAMVA-SUB-V-VIN-NUMBER PIC X(25) .
04 NY-AAMVA-DES-V-YEAR PIC X(4) .
04 NY-AAMVA-SUB-V-YEAR PIC X(4) .
04 NY-AAMVA-DES-V-MAKE PIC X(4) .
04 NY-AAMVA-SUB-V-MAKE PIC X(5) .
04 NY-AAMVA-DES-V-REPL-IND PIC X(4) .
04 NY-AAMVA-SUB-V-REPL-IND PIC X(1) .
04 NY-AAMVA-DES-V-HIST-IND PIC X(4) .
04 NY-AAMVA-SUB-V-HIST-IND PIC X(1) .
04 NY-AAMVA-DES-V-TOW-IND PIC X(4) .
04 NY-AAMVA-SUB-V-TOW-IND PIC X(1) .
04 NY-AAMVA-DES-VEH-END PIC X(1) .
02 NY-AAMVA-INSURANCE-SUBFILE .
04 NY-AAMVA-SUB-INS-START PIC X(2) .
04 NY-AAMVA-DES-I-ISSUER-ID PIC X(4) .
04 NY-AAMVA-SUB-I-ISSUER-ID PIC X(10) .
04 NY-AAMVA-DES-I-INS-CO-CODE PIC X(4) .
04 NY-AAMVA-SUB-I-INS-CO-CODE PIC X(3) .
04 NY-AAMVA-DES-I-INS-ISSUE-DT PIC X(4) .
04 NY-AAMVA-SUB-I-INS-ISSUE-DT PIC X(8) .
04 NY-AAMVA-DES-I-COV-START-DT PIC X(4) .
04 NY-AAMVA-SUB-I-COV-START-DT PIC X(8) .
04 NY-AAMVA-DES-I-COV-END-DT PIC X(4) .
04 NY-AAMVA-SUB-I-COV-END-DT PIC X(8) .
04 NY-AAMVA-DES-I-POLICY-NUMBER PIC X(4) .
04 NY-AAMVA-SUB-I-POLICY-NUMBER PIC X(15) .
04 NY-AAMVA-DES-INS-END PIC X(1) .
02 NY-AAMVA-DIG-CERT-SUBFILE .
04 NY-AAMVA-SUB-SIG-START PIC X(2) .
04 NY-AAMVA-DES-S-NY-BARCODE PIC X(4) .
04 NY-AAMVA-SUB-S-NY-BARCODE PIC X(8) .
04 NY-AAMVA-DES-S-SIG-TYPE PIC X(4) .
04 NY-AAMVA-SUB-S-SIG-TYPE PIC X(3) .
04 NY-AAMVA-DES-S-DIG-SIG PIC X(4) .
04 NY-AAMVA-SUB-S-DIG-SIG PIC X(32) .
04 NY-AAMVA-DES-SIG-END PIC X(1) .

```

EXAMINING THE OUTPUT DATA

The NY-AAMVA-INSURANCE-INFO structure represents 570 characters of data. This table describes in more detail each of the data elements and its content or source.

NY-AAMVA-INSURANCE-INFO

| Output element | Offset | Length | Description |
|----------------------------|--------|--------|-------------|
| Header information | | | |
| NY-AAMVA-CMPL-IND | 1 | 1 | “\x40” |
| NY-AAMVA-DATA-SEP | 2 | 1 | “\x0A” |
| NY-AAMVA-RECORD-SEP | 3 | 1 | “\x1C” |
| NY-AAMVA-SEGMENT-TERM | 4 | 1 | “\x0D” |
| NY-AAMVA-FILE-TYPE | 5 | 5 | “AAMVA” |
| NY-AAMVA-IIN | 10 | 6 | “636001” |
| NY-AAMVA-VERSION-NUMBER | 16 | 2 | “03” |
| NY-AAMVA-NO-OF-SUBFILES | 18 | 2 | “04” |
| Sub-file information | | | |
| NY-AAMVA-SUB-REG-TYPE | 20 | 2 | “RG” |
| NY-AAMVA-REG-OFFSET | 22 | 4 | “0060” |
| NY-AAMVA-REG-LEN | 26 | 4 | “0310” |
| NY-AAMVA-SUB-VEH-TYPE | 30 | 2 | “VH” |
| NY-AAMVA-VEH-OFFSET | 32 | 4 | “0370” |
| NY-AAMVA-VEH-LEN | 36 | 4 | “0064” |
| NY-AAMVA-SUB-INS-TYPE | 40 | 2 | “FR” |
| NY-AAMVA-INS-OFFSET | 42 | 4 | “0434” |
| NY-AAMVA-INS-LEN | 46 | 4 | “0079” |
| NY-AAMVA-SUB-SIG-TYPE | 50 | 2 | “SI” |
| NY-AAMVA-SIG-OFFSET | 52 | 4 | “0513” |
| NY-AAMVA-SIG-LEN | 56 | 4 | “0058” |
| Registrant sub-file | | | |
| NY-AAMVA-SUB-REG-START | 60 | 2 | “RG” |
| NY-AAMVA-DES-R-LAST-NAME-1 | 62 | 4 | “\x0ARBD” |

| Output element | Offset | Length | Description |
|------------------------------|--------|--------|---------------------------------|
| NY-AAMVA-SUB-R-LAST-NAME-1 | 66 | 18 | From NYID-R-LAST-NAME-1 |
| NY-AAMVA-DES-R-GIVEN-NAME-1 | 84 | 4 | “\x0ARBE” |
| NY-AAMVA-SUB-R-GIVEN-NAME-1 | 88 | 16 | From NYID-R-FIRST-NAME-1 |
| NY-AAMVA-DES-R-MIDDLE-INIT-1 | 104 | 4 | “\x0ARBF” |
| NY-AAMVA-SUB-R-MIDDLE-INIT-1 | 108 | 1 | From NYID-R-MIDDLE-NAME-1 |
| NY-AAMVA-DES-R-NAME-SUFIX-1 | 109 | 4 | “\x0ARBG” |
| NY-AAMVA-SUB-R-NAME-SUFIX-1 | 113 | 3 | From NYID-R-NAME-SUFFIX-1 |
| NY-AAMVA-DES-R-NAME-AS-ONE-1 | 116 | 4 | “\x0ARBC” |
| NY-AAMVA-SUB-R-NAME-AS-ONE-1 | 120 | 20 | From NYID-R-NAME-AS-ONE-1 |
| NY-AAMVA-DES-R-CLIENT-ID-1 | 140 | 4 | “\x0ADAQ” |
| NY-AAMVA-SUB-R-CLIENT-ID-1 | 144 | 9 | From NYID-R-CLIENT-ID-1 |
| NY-AAMVA-DES-R-LAST-NAME-2 | 153 | 4 | “\x0ARBD” |
| NY-AAMVA-SUB-R-LAST-NAME-2 | 157 | 18 | From NYID-R-LAST-NAME-2 |
| NY-AAMVA-DES-R-GIVEN-NAME-2 | 175 | 4 | “\x0ARBE” |
| NY-AAMVA-SUB-R-GIVEN-NAME-2 | 179 | 16 | From NYID-R-FIRST-NAME-2 |
| NY-AAMVA-DES-R-MIDDLE-INIT-2 | 195 | 4 | “\x0ARBF” |
| NY-AAMVA-SUB-R-MIDDLE-INIT-2 | 199 | 1 | From NYID-R-MIDDLE-NAME-2 |
| NY-AAMVA-DES-R-NAME-SUFIX-2 | 200 | 4 | “\x0ARBG” |
| NY-AAMVA-SUB-R-NAME-SUFIX-2 | 204 | 3 | From NYID-R-NAME-SUFFIX-2 |
| NY-AAMVA-DES-R-NAME-AS-ONE-2 | 207 | 4 | “\x0ARBC” |
| NY-AAMVA-SUB-R-NAME-AS-ONE-2 | 211 | 20 | From NYID-R-NAME-AS-ONE-2 |
| NY-AAMVA-DES-R-CLIENT-ID-2 | 231 | 4 | “\x0ADAQ” |
| NY-AAMVA-SUB-R-CLIENT-ID-2 | 235 | 9 | From NYID-R-CLIENT-ID-2 |
| NY-AAMVA-DES-R-ORG-LINE-1 | 244 | 4 | “\x0AZBC” |
| NY-AAMVA-SUB-R-ORG-LINE-1 | 248 | 20 | From NYID-R-ORGANIZATION-LINE-1 |
| NY-AAMVA-DES-R-ORG-LINE-2 | 268 | 4 | “\x0AZBD” |
| NY-AAMVA-SUB-R-ORG-LINE-2 | 272 | 20 | From NYID-R-ORGANIZATION-LINE-2 |
| NY-AAMVA-DES-R-ORG-FEIN | 292 | 4 | “\x0AZBE” |

| Output element | Offset | Length | Description |
|-------------------------|--------|--------|-------------------------------|
| NY-AAMVA-SUB-R-ORG-FEIN | 296 | 9 | From NYID-R-ORGANIZATION-FEIN |
| NY-AAMVA-DES-R-SEATS | 305 | 4 | “\x0ARAP” |
| NY-AAMVA-SUB-R-SEATS | 309 | 2 | From NYID-R-V-SEATS |
| NY-AAMVA-DES-R-STREET | 311 | 4 | “\x0ARBN” |
| NY-AAMVA-SUB-R-STREET | 315 | 20 | From NYID-R-STREET |
| NY-AAMVA-DES-R-CITY | 335 | 4 | “\x0ARBP” |
| NY-AAMVA-SUB-R-CITY | 339 | 15 | From NYID-R-CITY |
| NY-AAMVA-DES-R-STATE | 354 | 4 | “\x0ARBQ” |
| NY-AAMVA-SUB-R-STATE | 358 | 2 | From NYID-R-STATE |
| NY-AAMVA-DES-R-ZIP-CODE | 360 | 4 | “\x0ARBR” |
| NY-AAMVA-SUB-R-ZIP-CODE | 364 | 5 | From NYID-R-ZIP-CODE |
| NY-AAMVA-DES-REG-END | 369 | 1 | “\x0D” |

Vehicle sub-file

| | | | |
|---------------------------|-----|----|------------------------|
| NY-AAMVA-SUB-VEH-START | 370 | 2 | “VH” |
| NY-AAMVA-DES-V-VIN-NUMBER | 372 | 4 | “\x0AVAD” |
| NY-AAMVA-SUB-V-VIN-NUMBER | 376 | 25 | From NYID-V-VIN-NUMBER |
| NY-AAMVA-DES-V-YEAR | 401 | 4 | “\x0AVAL” |
| NY-AAMVA-SUB-V-YEAR | 405 | 4 | From NYID-V-YEAR |
| NY-AAMVA-DES-V-MAKE | 409 | 4 | “\x0AVAK” |
| NY-AAMVA-SUB-V-MAKE | 413 | 5 | From NYID-V-MAKE |
| NY-AAMVA-DES-V-REPL-IND | 418 | 4 | “\x0AZZD” |
| NY-AAMVA-SUB-V-REPL-IND | 422 | 1 | From NYID-V-REP-IND |
| NY-AAMVA-DES-V-HIST-IND | 423 | 4 | “\x0AZZE” |
| NY-AAMVA-SUB-V-HIST-IND | 427 | 1 | From NYID-V-HIST-IND |
| NY-AAMVA-DES-V-TOW-IND | 428 | 4 | “\x0AZZF” |
| NY-AAMVA-SUB-V-TOW-IND | 432 | 1 | From NYID-V-TOW-IND |
| NY-AAMVA-DES-VEH-END | 433 | 1 | “\x0D” |

Insurance sub-file

| Output element | Offset | Length | Description |
|------------------------------|--------|--------|---------------------------------|
| NY-AAMVA-SUB-INS-START | 434 | 2 | “FR” |
| NY-AAMVA-DES-I-ISSUER-ID | 436 | 4 | “\x0AFAA” |
| NY-AAMVA-SUB-I-ISSUER-ID | 440 | 10 | From NYID-I-ISSUER-ID |
| NY-AAMVA-DES-I-INS-CO-CODE | 450 | 4 | “\x0AZZC” |
| NY-AAMVA-SUB-I-INS-CO-CODE | 454 | 3 | From NYID-I-INS-COMPANY-CODE |
| NY-AAMVA-DES-I-INS-ISSUE-DT | 457 | 4 | “\x0AFAB” |
| NY-AAMVA-SUB-I-INS-ISSUE-DT | 461 | 8 | From NYID-I-INS-ISSUANCE-DATE |
| NY-AAMVA-DES-I-COV-START-DT | 469 | 4 | “\x0AFAC” |
| NY-AAMVA-SUB-I-COV-START-DT | 473 | 8 | From NYID-I-COVERAGE-START-DATE |
| NY-AAMVA-DES-I-COV-END-DT | 481 | 4 | “\x0AFAD” |
| NY-AAMVA-SUB-I-COV-END-DT | 485 | 8 | From NYID-I-COVERAGE-END-DATE |
| NY-AAMVA-DES-I-POLICY-NUMBER | 493 | 4 | “\x0AZZB” |
| NY-AAMVA-SUB-I-POLICY-NUMBER | 497 | 15 | From NYID-I-POLICY-NUMBER |
| NY-AAMVA-DES-INS-END | 512 | 1 | “\x0D” |

Signature sub-file

| | | | |
|---------------------------|-----|----|-------------------------------|
| NY-AAMVA-SUB-SIG-START | 513 | 2 | “SI” |
| NY-AAMVA-DES-S-NY-BARCODE | 515 | 4 | “\x0AZZZ” |
| NY-AAMVA-SUB-S-NY-BARCODE | 519 | 8 | “IC200010” |
| NY-AAMVA-DES-S-SIG-TYPE | 527 | 4 | “\x0ASAA” |
| NY-AAMVA-SUB-S-SIG-TYPE | 531 | 3 | “001” |
| NY-AAMVA-DES-S-DIG-SIG | 534 | 4 | “\x0ASAB” |
| NY-AAMVA-SUB-S-DIG-SIG | 538 | 32 | From NYID-S-DIGITAL-SIGNATURE |
| NY-AAMVA-DES-SIG-END | 570 | 1 | “\0x0D” |

MAPPING INPUT DATA TO OUTPUT DATA

The NY-INSURANCE-INFO structure is the source of the transaction data that will define the output data in the NY-AAMVA-INSURANCE-INFO structure. A subset of the data elements is moved to the output structure. In some cases, the source data is truncated in the output structure.

This table details which data elements will be transferred to the output structure (and thereby the bar code). This will help define which data elements are actually necessary to assign in the application to create a NYSID bar code.

| Source record | Length | Destination record | Length | Required |
|----------------------------|--------|------------------------------|--------|----------|
| NYID-R-LAST-NAME-1 | 18 | NY-AAMVA-SUB-R-LAST-NAME-1 | 18 | Yes |
| NYID-R-FIRST-NAME-1 | 16 | NY-AAMVA-SUB-R-GIVEN-NAME-1 | 16 | Yes |
| NYID-R-MIDDLE-NAME-1 | 16 | NY-AAMVA-SUB-R-MIDDLE-INIT-1 | 1 | No |
| NYID-R-NAME-SUFFIX-1 | 3 | NY-AAMVA-SUB-R-NAME-SUFIX-1 | 3 | No |
| NYID-R-NAME-AS-ONE-1 | 20 | NY-AAMVA-SUB-R-NAME-AS-ONE-1 | 20 | Yes * |
| NYID-R-CLIENT-ID-1 | 9 | NY-AAMVA-SUB-R-CLIENT-ID-1 | 9 | Yes |
| NYID-R-LAST-NAME-2 | 18 | NY-AAMVA-SUB-R-LAST-NAME-2 | 18 | No |
| NYID-R-FIRST-NAME-2 | 16 | NY-AAMVA-SUB-R-GIVEN-NAME-2 | 16 | No |
| NYID-R-MIDDLE-NAME-2 | 16 | NY-AAMVA-SUB-R-MIDDLE-INIT-2 | 1 | No |
| NYID-R-NAME-AS-ONE-2 | 20 | NY-AAMVA-SUB-R-NAME-AS-ONE-2 | 20 | No |
| NYID-R-CLIENT-ID-2 | 9 | NY-AAMVA-SUB-R-CLIENT-ID-2 | 9 | No |
| NYID-R-ORGANIZATION-LINE-1 | 20 | NY-AAMVA-SUB-R-ORG-LINE-1 | 20 | Yes |
| NYID-R-ORGANIZATION-LINE-2 | 20 | NY-AAMVA-SUB-R-ORG-LINE-2 | 20 | No |
| NYID-R-ORGANIZATION-FEIN | 9 | NY-AAMVA-SUB-R-ORG-FEIN | 9 | Yes |
| NYID-R-V-SEATS | 2 | NY-AAMVA-SUB-R-SEATS | 2 | No** |
| NYID-R-STREET | 20 | NY-AAMVA-SUB-R-STREET | 20 | Yes |
| NYID-R-CITY | 15 | NY-AAMVA-SUB-R-CITY | 15 | Yes |
| NYID-R-STATE | 2 | NY-AAMVA-SUB-R-STATE | 2 | Yes |
| NYID-R-ZIP-CODE | 9 | NY-AAMVA-SUB-R-ZIP-CODE | 5 | Yes |
| NYID-V-VIN-NUMBER | 25 | NY-AAMVA-SUB-V-VIN-NUMBER | 25 | Yes |
| NYID-V-YEAR | 4 | NY-AAMVA-SUB-V-YEAR | 4 | Yes |

* If you omit data from this field, it will be added by software routines provided by the New York State DMV.

**The number of seats is a required field for FH-1 and FH-1B ID cards issued to For Hire vehicles (taxis, liveries, rentals, school cars, buses, and so on).

| Source record | Length | Destination record | Length | Required |
|-------------------------------|--------|------------------------------|--------|----------|
| NYID-V-MAKE | 5 | NY-AAMVA-SUB-V-MAKE | 5 | Yes |
| NYID-V-REP-IND | 1 | NY-AAMVA-SUB-V-REP-IND | 1 | Yes |
| NYID-V-HIST-IND | 1 | NY-AAMVA-SUB-V-HIST-IND | 1 | No |
| NYID-V-TOW-IND | 1 | NY-AAMVA-SUB-V-TOW-IND | 1 | No |
| NYID-I-ISSUER-ID | 10 | NY-AAMVA-SUB-I-ISSUER-ID | 10 | Yes |
| NYID-I-INS-COMPANY-CODE | 3 | NY-AAMVA-SUB-I-INS-CO-CODE | 3 | Yes |
| NYID-I-INS-ISSUANCE-DATE | 8 | NY-AAMVA-SUB-I-INS-ISSUE-DT | 8 | Yes |
| NYID-I-COVERAGE-START-DATE | 8 | NY-AAMVA-SUB-I-COV-START-DT | 8 | Yes |
| NYID-I-COVERAGE-END-DATE | 8 | NY-AAMVA-SUB-I-COV-END-DT | 8 | Yes |
| NYID-I-POLICY-NUMBER | 15 | NY-AAMVA-SUB-I-POLICY-NUMBER | 15 | Yes |
| Hard code: "IC200010" | | NY-AAMVA-SUB-S-NY-BARCODE | 8 | No |
| NYID-S-SIGNATURE-TYPE ("001") | 4 | NY-AAMVA-SUB-S-SIG-TYPE | 3 | Yes |
| NYID-S-DIGITAL-SIGNATURE | 32 | NY-AAMVA-SUB-S-DIG-SIG | 32 | Yes |
| NYID-V-VIN-OVERRIDE | 1 | Not used in the bar code | - | - |
| NYID-I-DOCUMENT-TYPE | 15 | Not used in the bar code | - | - |
| NYID-I-PAPER-SELECTION | 15 | Not used in the bar code | - | - |
| NYID-I-INS-COMPANY-NAME | 40 | Not used in the bar code | - | - |
| NYID-I-AGENCY-NAME | 40 | Not used in the bar code | - | - |
| NYID-I-AGENCY-ADDRESS-LINE-1 | 40 | Not used in the bar code | - | - |
| NYID-I-AGENCY-ADDRESS-LINE-2 | 40 | Not used in the bar code | - | - |
| NYID-I-IS-ORGANIZATION | 1 | Not used in the bar code | - | - |

* If you omit data from this field, it will be added by software routines provided by the New York State DMV.

**The number of seats is a required field for FH-1 and FH-1B ID cards issued to For Hire vehicles (taxis, liveries, rentals, school cars, buses, and so on).

INPUT DATA FORMATTING RULES

Some of the data fields in the input structure have special formatting requirements specified by the NYSID.

These requirements are listed below.

| Source record | Length | Destination record | Length | Required |
|----------------------------|--------|------------------------------|--------|----------|
| NYID-R-NAME-AS-ONE-1 | 20 | NY-AAMVA-SUB-R-NAME-AS-ONE-1 | 20 | Yes |
| NYID-R-NAME-AS-ONE-2 | 20 | NY-AAMVA-SUB-R-NAME-AS-ONE-2 | 20 | No |
| NYID-R-ORGANIZATION-LINE-1 | 20 | NY-AAMVA-SUB-R-ORG-LINE-1 | 20 | Yes |
| NYID-R-ORGANIZATION-LINE-2 | 20 | NY-AAMVA-SUB-R-ORG-LINE-2 | 20 | No |

Format the source data with “@” to separate each field as follows:

“LASTNAME@FIRSTNAME@MI”

Format the organization name using “@” character to separate each word. The name can overflow into the second line. For example:

“JOHN@SMITH@DBA@SMITH@TRUCKING”

NOTE: Supply the input dates in the InsuranceInfo structure (which is passed to the PNYLoadAAMVA API) in this format: *MMDDYYYY*.

The New York routines, when moving information from the InsuranceInfo structure to the InsuranceAAMVA structure, convert these dates into *YYYYMMDD* format. This format swapping is built into those routines.

PDF417 FXR FILES

The PDF417 resources you receive from Oracle Insurance for Documaker include FXR files you can use. Prior to version 11.2, the PDF417_2.FXR and PDF417.FXR files were provided to licensed PDF417 customers. Included in version 11.2, the REL112.FXR file contains the PDF417 font references found in the two PDF417 FXR files previously included in this product.

The 0912 and 1216 font IDs from the PDF417.FXR file (used for 300 dpi printing) are included in the REL112.FXR file.

The 0912 and 1216 font IDs from the PDF417_2.FXR file (only used if your primary printer is an AFP 240 dpi printer) are included in the REL112.FXR file as font IDs 0911 and 1215 to avoid conflicting with the 0912 and 1216 font IDs used for 300 dpi printing.

This table summarizes the differences in the FXR files:

| FXR file | Used in version | For these printers | Contains these font IDs |
|--------------|-----------------|--------------------|--|
| PDF417.FXR | 11.1 or lower | 300 dpi | 0912 and 1216 |
| PDF417_2.FXR | 11.1 or lower | 240 dpi AFP only | 0912 and 1216 |
| REL112.FXR | 11.2 and higher | 300 or 240 dpi | 0912 and 1216 (300 dpi) 0911 and 1215 (240 dpi) |

If you are running version 11.1 or lower:

- Only use the PDF417_2.FXR file when your primary printer is an AFP 240 dpi printer. You can also use this FXR file to print to other printers, but only if your primary printer is an AFP 240 dpi printer.
- Use the PDF417.FXR file in all other cases. For instance, if your primary printer is a PCL, Metacode, or 300 dpi AFP printer, you would use the PDF417.FXR file.

If you are running version 11.2 or higher, use the REL112.FXR file.

You can include the contents of these FXR files into the FXR file you are currently using. The Font Manager lets you insert fonts from another FXR file. For more information, see the information on inserting fonts in the section, Using the Font Manager, in the Docucreate User Guide.

NOTE: If the font IDs conflict with existing font IDs, renumber them.

PDF417 FONTS

There are 16 characters represented in each of the fonts provided. Each font contains bitmaps for the character set consisting of the underscore character and the uppercase letters A through O.

The characters are fixed pitch. Each cell is evenly divisible by four, to represent four bar-space pattern strokes. The cell height establishes a three-to-one ratio over the stroke width.

Oracle offers two font character cell sizes:

- 12 dots wide by 9 dots tall (approximately a 2 point font size)
- 16 dots wide by 12 dots tall (approximately a 3 point font size)

Oracle offers bitmap fonts for the following printer families:

- AFP (both 240dpi and 300dpi)
- Metacode 300dpi (both portrait and landscape)
- PCL 300dpi (PCL Level 5 (LJ3) and up)

Oracle also offers scalable vector TrueType and PostScript fonts.

| Printer family | Cell dimensions | Printer resolution | Page orientation | Font files provided |
|-----------------------|--|--------------------|------------------|--|
| PostScript Type One | Specify one of these point sizes: 2.16pt or 2.88pt | Not applicable | Not applicable | PDF417__.PFB |
| TrueType (RP runtime) | Specify one of these point sizes: 2.16pt or 2.88pt | Not applicable | Not applicable | PDF417__.TTF |
| AFP | 9 H x 12 W | 240 dpi | Standard | Char set: C0P09X12.240 Coded font: X0P09X12.FNT Code page: T1DOC037. |
| AFP | 12 H x 16 W | 240 dpi | Standard | Char set: C0P12X16.240 Coded font: X0P12X16.FNT Code page: T1DOC037. |
| AFP | 9 H x 12 W | 300 dpi | Standard | Char set: C0P09X12.300 Coded font: X0P09X12.FNT Code page: T1DOC037. |
| AFP | 12 H x 16 W | 300 dpi | Standard | Char set: C0P12X16.300 Coded font: X0P12X16.FNT Code page: T1DOC037. |
| AFP | 15 H x 20 W | 300 dpi | Standard | Char set: C0P15X20.300 Coded font: X0P15X20.FNT Code page: T1DOC037 |

| Printer family | Cell dimensions | Printer resolution | Page orientation | Font files provided |
|----------------|-----------------|--------------------|------------------|---------------------|
| Metacode | 9 H x 12 W | 300 dpi | Portrait | P09X12.FNT |
| Metacode | 12 H x 16 W | 300 dpi | Portrait | P12X16.FNT |
| Metacode | 15 H x 20 W | 300 dpi | Portrait | P15X20.FNT |
| Metacode | 9 H x 12 W | 300 dpi | Landscape | L09X12.FNT |
| Metacode | 12 H x 16 W | 300 dpi | Landscape | L12X16.FNT |
| Metacode | 15 H x 20 W | 300 dpi | Landscape | L15X20.FNT |
| PCL | 9 H x 12 W | 300 dpi | Standard | P09X12.PCL |
| PCL | 12 H x 16 W | 300 dpi | Standard | P12X16.PCL |
| PCL | 15 H x 20 W | 300 dpi | Standard | P15X20.PCL |

The 15x20 fonts, with the exception of the AFP 300 dpi font, are only referenced in the PDF417_2.FXR file in version 11.1 or lower.

NOTE: Prior to version 11.2, the PDF417_2.FXR and PDF417.FXR files were provided to licensed PDF417 customers. Included in version 11.2, the REL112.FXR file contains the PDF417 font references found in two PDF417 FXR files previously included in the product.

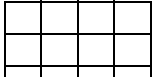
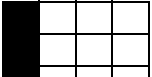
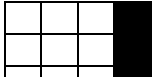
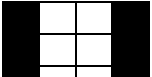




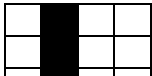
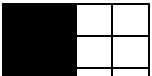
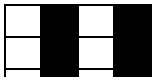





The 0912 and 1216 font IDs from the PDF417.FXR file (used for 300 dpi printing) are included in the REL112.FXR file. The 0912 and 1216 font IDs from the PDF417_2.FXR file (only used if your primary printer is an AFP 240 dpi printer) are included in the REL112.FXR file as font IDs 0911 and 1215 to avoid conflicting with the 0912 and 1216 font IDs used for 300 dpi printing.

Only use the PDF417_2.FXR file when your primary printer is an AFP 240 dpi printer. You can use this FXR to print to other printers (300 dpi) as well but only use it when your primary printer is an AFP 240 dpi printer.

NOTE: This is accomplished by using a different font when printing to a 300 dpi printer. For example, font ID 912 uses a 9x12 font when printing to an AFP 240 dpi printer but uses the 12x16 font when printing to 300 dpi printers (PCL, Metacode). Font ID 1216 uses a 12x16 font when printing to an AFP 240 dpi printer but uses the 15x20 font when printing to 300 dpi printers (PCL, Metacode).

**PDF417
CHARACTER
SET**

Here is a representation of the Oracle font character set for PDF417.

| Character | Character |
|---|---|
| Underscore (_) | H |
|  |  |
| A | I |
|  |  |
| B | J |
|  |  |
| C | K |
|  |  |
| D | L |
|  |  |
| E | M |
|  |  |
| F | N |
|  |  |
| G | O |
|  |  |

SIZING A PDF417 BAR CODE

A PDF417 bar code can represent a limited amount of data. This data is converted internally and stored in what are called *code words*. There is a maximum of 928 internal code words per bar code. Some of the code words are dedicated to the data content itself, and some of the code words are usually dedicated to error correction (ECC) encoding.

The ECC level can range between zero (0) and eight (8). Each increase in level requires an increase in the amount of the bar code's code words dedicated to error correction. Since there is a maximum number of code words allowed, increasing the amount of error correction decreases the maximum space available for data.

The number of code words dedicated to error correction is calculated in this fashion:

$$CW = 2 ** (ECC + 1)$$

For example:

$$\begin{aligned} \text{ECC} = 0, \text{ CW} &= 2 ** (0+1) = 2 ** 1 = 2 \\ \text{ECC} = 1, \text{ CW} &= 2 ** (1+1) = 2 ** 2 = 4 \\ \text{ECC} = 2, \text{ CW} &= 2 ** (2+1) = 2 ** 3 = 8 \\ \text{ECC} = 3, \text{ CW} &= 2 ** (3+1) = 2 ** 4 = 16 \\ \text{ECC} = 4, \text{ CW} &= 2 ** (4+1) = 2 ** 5 = 32 \\ \text{ECC} = 5, \text{ CW} &= 2 ** (5+1) = 2 ** 6 = 64 \\ \text{ECC} = 6, \text{ CW} &= 2 ** (6+1) = 2 ** 7 = 128 \\ \text{ECC} = 7, \text{ CW} &= 2 ** (7+1) = 2 ** 8 = 256 \\ \text{ECC} = 8, \text{ CW} &= 2 ** (8+1) = 2 ** 9 = 512 \end{aligned}$$

The NYSID requires an ECC level of at least 4, equating to 32 internal code words.

The total number of code words (up to the maximum of 928) that can be contained in a PDF417 bar code is computed by determining how many code words are represented by the number of columns in each row, and then multiplying by the number of rows.

The number of rows can range from 3 to 90.

The number of code words per row can be calculated from the specified columns using the following formula:

$$CW = \text{INT}((\text{COLS} * 100) + 425) / 425 - 5$$

The resultant code words per row can range from 1 to 30. This puts a practical range on the number of columns per row of from 22 to 145.

There is an absolute limit to the size of a bar code of 928 internal code words. All input data and error correction information must be able to be represented by no more than 928 internal code words. Generally, two code words are always consumed by overhead.

The bar code application for the state of New York currently stores 570 characters of text into the bar code. A PDF417 bar code will compact the data somewhat. The fastest and safest method of encoding the data is called *binary mode*. Binary mode compacts the data by approximately a factor of 1.2. For example:

$$\begin{aligned} 570 / 1.2 &= 475 \text{ data code words} \\ \text{ECC level } 4 &= 32 \text{ code words} \\ \text{Overhead} &= 2 \text{ code words} \\ \text{Total required} &= 509 \text{ code words} \end{aligned}$$

There are many size and shape combinations of PDF417 bar codes that are just large enough to represent this information. By multiplying a ROWS value times a code words per row value, it is possible to find many combinations greater than 509.

This table shows how to equate text columns to code words per row.

| Text columns | Internal code words |
|--------------|---------------------|
| 149 and more | Too big |
| 145-148 | 30 |
| 141-144 | 29 |
| 136-139 | 28 |
| 132-135 | 27 |
| 128-131 | 26 |
| 124-127 | 25 |
| 119-123 | 24 |
| 115-118 | 23 |
| 111-114 | 22 |
| 107-110 | 21 |
| 102-106 | 20 |
| 98-101 | 19 |
| 94-97 | 18 |
| 90-93 | 17 |
| 85-89 | 16 |
| 81-84 | 15 |
| 77-80 | 14 |
| 73-76 | 13 |
| 68-72 | 12 |
| 64-67 | 11 |
| 60-63 | 10 |
| 56-59 | 9 |
| 51-55 | 8 |
| 47-50 | 7 |
| 43-46 | 6 |
| 39-42 | 5 |

| Text columns | Internal code words |
|--------------|---------------------|
| 34-38 | 4 |
| 30-33 | 3 |
| 26-29 | 2 |
| 22-25 | 1 |
| 21 and less | Too small |

For example 30 rows times 18 code words per row equals 540, which should be large enough. And, 18 code words per row equate to 94 text columns. So, a bar code of 30 rows and 94 columns should be safely large enough for the New York application.

From the previous tables, assuming the smaller font and 300 dpi, the bar code would be 9/10ths of an inch tall and about 3 and 7/10ths inches wide.

DETERMINING THE COLUMNS PER ROW

Use this table to determine the number of font character columns that will be required to create a bar code of the desired width. Notice that because of the way the PDF417 bar code works, only certain numbers of columns can be returned.

Programs should be written to use one of the choices listed in the table. If the program requests a different size, the remaining columns at the end of the row will be padded with underscores.

For example, if the program establishes a row of one hundred character columns in width, the Oracle API returns 98 columns of bar code data and fills the remaining characters at the end of each row with underscores.

| If each row in the receiving buffer is this many bytes (PIC X(39)) | Here is the resulting bar code width in inches, using a... | | | |
|--|--|--|--|--|
| | 12-dot character width font at 300 dpi | 12-dot character width font at 240 dpi | 16-dot character width font at 300 dpi | 16-dot character width font at 240 dpi |
| 39-42 | 1.52 | 1.90 | 2.03 | 2.53 |
| 43-46 | 1.68 | 2.10 | 2.24 | 2.80 |
| 47-50 | 1.88 | 2.35 | 2.51 | 3.13 |
| 51-55 | 2.04 | 2.55 | 2.72 | 3.40 |
| 56-59 | 2.20 | 2.75 | 2.93 | 3.67 |
| 60-63 | 2.36 | 2.95 | 3.15 | 3.93 |
| 64-67 | 2.56 | 3.20 | 3.41 | 4.27 |
| 68-72 | 2.72 | 3.40 | 3.63 | 4.53 |
| 73-76 | 2.88 | 3.60 | 3.84 | 4.80 |
| 77-80 | 3.04 | 3.80 | 4.05 | 5.07 |
| 81-84 | 3.24 | 4.05 | 4.32 | 5.40 |
| 85-89 | 3.40 | 4.25 | 4.53 | 5.67 |
| 90-93 | 3.56 | 4.45 | 4.75 | 5.93 |
| 94-97 | 3.72 | 4.65 | 4.96 | 6.20 |
| 98-101 | 3.92 | 4.90 | 5.23 | 6.53 |
| 102-106 | 4.08 | 5.10 | 5.44 | 6.80 |
| 107-110 | 4.24 | 5.30 | 5.65 | 7.07 |
| 111-114 | 4.40 | 5.50 | 5.87 | 7.33 |
| 115-118 | 4.60 | 5.75 | 6.13 | 7.67 |
| 119-123 | 4.76 | 5.95 | 6.35 | 7.93 |

| If each row in the receiving buffer is this many bytes (PIC X(39)) | Here is the resulting bar code width in inches, using a... | | | |
|--|--|--|--|--|
| | 12-dot character width font at 300 dpi | 12-dot character width font at 240 dpi | 16-dot character width font at 300 dpi | 16-dot character width font at 240 dpi |
| 124-127 | 4.92 | 6.15 | 6.56 | 8.20 |
| 128-131 | 5.08 | 6.35 | 6.77 | 8.47 |
| 132-135 | 5.28 | 6.60 | 7.04 | 8.80 |

DETERMINING THE NUMBER OF ROWS

Use this table as a guideline to determine the number of font character rows that will be required to create a bar code of the desired height.

Other values are allowed, and the table can be a guide on extrapolating other row sizes. The PDF417 bar code specification has a maximum limit of 90 rows.

| For each row in the receiving buffer (OCCURS 20 TIMES) | Here is the resulting bar code height in inches, using a... | | | |
|--|---|--|---|---|
| | 9-dot character height font at 300 dpi | 9-dot character height font at 240 dpi | 12-dot character height font at 300 dpi | 12-dot character height font at 240 dpi |
| 10 | 0.3000 | 0.3750 | 0.4000 | 0.5000 |
| 11 | 0.3300 | 0.4125 | 0.4400 | 0.5500 |
| 12 | 0.3600 | 0.4500 | 0.4800 | 0.6000 |
| 13 | 0.3900 | 0.4875 | 0.5200 | 0.6500 |
| 14 | 0.4200 | 0.5250 | 0.5600 | 0.7000 |
| 15 | 0.4500 | 0.5625 | 0.6000 | 0.7500 |
| 16 | 0.4800 | 0.6000 | 0.6400 | 0.8000 |
| 17 | 0.5100 | 0.6375 | 0.6800 | 0.8500 |
| 18 | 0.5400 | 0.6750 | 0.7200 | 0.9000 |
| 19 | 0.5700 | 0.7125 | 0.7600 | 0.9500 |
| 20 | 0.6000 | 0.7500 | 0.8000 | 1.0000 |
| 21 | 0.6300 | 0.7875 | 0.8400 | 1.0500 |
| 22 | 0.6600 | 0.8250 | 0.8800 | 1.1000 |
| 23 | 0.6900 | 0.8625 | 0.9200 | 1.1500 |
| 24 | 0.7200 | 0.9000 | 0.9600 | 1.2000 |
| 25 | 0.7500 | 0.9375 | 1.0000 | 1.2500 |
| 26 | 0.7800 | 0.9750 | 1.0400 | 1.3000 |
| 27 | 0.8100 | 1.0125 | 1.0800 | 1.3500 |
| 28 | 0.8400 | 1.0500 | 1.1200 | 1.4000 |
| 29 | 0.8700 | 1.0875 | 1.1600 | 1.4500 |
| 30 | 0.9000 | 1.1250 | 1.2000 | 1.5000 |
| 31 | 0.9300 | 1.1625 | 1.2400 | 1.5500 |
| 32 | 0.9600 | 1.2000 | 1.2800 | 1.6000 |

For each row in the receiving buffer (OCCURS 20 TIMES) Here is the resulting bar code height in inches, using a...

| | 9-dot character height font at 300 dpi | 9-dot character height font at 240 dpi | 12-dot character height font at 300 dpi | 12-dot character height font at 240 dpi |
|----|--|--|---|---|
| 33 | 0.9900 | 1.2375 | 1.3200 | 1.6500 |
| 34 | 1.0200 | 1.2750 | 1.3600 | 1.7000 |
| 35 | 1.0500 | 1.3125 | 1.4000 | 1.7500 |
| 36 | 1.0800 | 1.3500 | 1.4400 | 1.8000 |
| 37 | 1.1100 | 1.3875 | 1.4800 | 1.8500 |
| 38 | 1.1400 | 1.4250 | 1.5200 | 1.9000 |
| 39 | 1.1700 | 1.4625 | 1.5600 | 1.9500 |
| 40 | 1.2000 | 1.5000 | 1.6000 | 2.0000 |
| 41 | 1.2300 | 1.5375 | 1.6400 | 2.0500 |
| 42 | 1.2600 | 1.5750 | 1.6800 | 2.1000 |
| 43 | 1.2900 | 1.6125 | 1.7200 | 2.1500 |
| 44 | 1.3200 | 1.6500 | 1.7600 | 2.2000 |
| 45 | 1.3500 | 1.6875 | 1.8000 | 2.2500 |

TIPS This section provides information you may need to get the best results and to resolve any problems which may occur.

PRODUCING RELIABLE BAR CODES

The most reliable bar codes are produced when you use one of Documaker's standard print drivers (AFP, Xerox Metacode, PCL, or PostScript) to produce a print stream that is sent directly to the appropriate printer. Bar codes scanned directly from these printouts are more reliable when scanned because...

- Documaker completely controls the print stream produced
- The printouts are of high resolution (240 or 300 dots per inch)
- Most AFP, Xerox Metacode, PCL, or PostScript printers are very reliable and consistent

You lessen the reliability of scanned bar codes when you...

- Produce bar codes in an alternate document format such as PDF, RTF, or HTML
- Fax the bar code

Producing bar codes in PDF, RTF, or HTML format

When you produce bar codes in an alternate document format such as PDF, RTF, or HTML, you reduce the reliability of the bar codes being produced. Applications such as Internet Explorer, Microsoft Word, and Adobe Acrobat must first interpret and display the document format. The interpretation and display of the document will often vary depending upon the version of the software used. These applications rely on other applications, such as Windows print drivers, to print the document which includes the PDF417 bar codes. High reliability and consistency can be more difficult to achieve in this environment as there are many factors that affect the final printed results.

Faxing

Most fax machines scan and transmit documents at a far lower resolution than the original printed documents. High resolution is defined as 196 x 204 dpi and low resolution is defined as 96 x 104 dpi (dots per inch).

Improving readability

Here are some things you can do to improve the readability of PDF417 bar codes:

- If possible, increase the Error Correction (ECC) Encoding when producing the PDF417 bar code.
- If you must fax the document, fax at the highest possible resolution.
- Make sure you do not have any fit to width options enabled on your Window's print or FAX driver. These options shrink output and that can distort the bar code.
- Turn off any graphic smoothing features. Smoothing is used by some print or FAX drivers to make bitmaps look better. This can distort the bar code.
- If your FAX driver will accept PCL or PostScript output, try sending PCL or PostScript output to eliminate the PDF step.
- Use the option to print as a graphic image if the application or Windows print driver allows it. This approach is slower but may give you a more precise printout.

- Experiment with various advanced print settings within the 3rd party application and test using different printers and Windows print drivers.

TURNING ON TRACING

To help you spot errors, you can turn on tracing. To turn on tracing for the PNYLIB and PSYLIB modules, set the INI options shown below. Setting one or both of these options to Yes tells the system to create a trace file named PDF417.LOG.

```
< Debug_Switches >
  Enable_Debug_Options = Yes
  PNYLIB = Yes
  PSYLIB = Yes
```

NOTE: Omit these options or set them to No during normal operation. Tracing slows performance.

ERROR MESSAGES

Here is a summary of the error messages you may encounter.

PNY01A This table shows the error messages that might possibly be returned in the error return-code variable parameter passed to the PNY01A API that prepares the bar code data into the AAMVA-compliant format.

| Error | Description |
|-------|---|
| 0 | Successful completion. |
| 8 | Error in creating the document signature. Check the KEY and PASS PHRASE parameters. |

P4172FNT This table shows the error messages that might possibly be returned in the error return-code variable parameter passed to the P4172FNT API that creates the font text for a bar code.

The errors that have descriptions denoted by ***INTERNAL ERROR* should ordinarily never be observed by an application invoking the Oracle API. These errors should be reported to Oracle Support. The other errors may result from incorrect usage, and the description notes where to begin to determine the cause of the error.

| Error | Symbolic name | Description |
|-------|------------------------|--|
| 0 | ERR_NOERROR | Successful completion. |
| -4 | ERR_NULLDATASOURCE | **INTERNAL ERROR. No input routine was specified. |
| -5 | ERR_INVALIDINPUTOBJECT | **INTERNAL ERROR. The PDF object is corrupt. |
| -7 | ERR_TOOMANYCW | **INTERNAL ERROR. Re-size of bar code cannot contain data. |

| Error | Symbolic name | Description |
|-------|------------------------|--|
| -8 | ERR_NOTINPUTDEVICE | **INTERNAL ERROR. The PDF object was not input type. |
| -9 | ERR_OUTPUTDEVICE | **INTERNAL ERROR. Error initializing output object -- invalid type or memory allocation failure. |
| -10 | ERR_NOFILTERSELECTED | **INTERNAL ERROR. No output driver setup/installed. |
| -11 | ERR_TOOMUCHDATA | The input data length exceeds the bar code. The bar code size requested (via the rows and columns parameters) is not large enough to contain the input data. Check the form design and see if you can make the bar code area larger. |
| -12 | ERR_NOACTIVEBARCODE | **INTERNAL ERROR. Attempt to print/query without bar code. |
| -14 | ERR_MPCBTOOMUCH | ** INTERNAL ERROR. The bar code is too small for the MPDF information. |
| -16 | ERR_TOOMANYMPDF | **INTERNAL ERROR. There are too many bar codes for the MPDF information. |
| -17 | ERR_ECCEXCEEDSCAPACITY | The bar code is too small for the ECC specified. The bar code size requested (via the rows and columns parameters) might be large enough to contain the data, but is not large enough to represent the data given the error correction value you specified. Increase the size of the bar code or make sure the ECCLevel parameter is correct. |
| -20 | ERR_MEMORYALLOC | **INTERNAL ERROR. Failed to allocate a temporary buffer. |
| -21 | ERR_ARGUMENT | **INTERNAL ERROR. Invalid argument: not supplied, out of range, and so on. If this occurs, make sure the input parameters are valid. |
| -22 | ERR_INVALIDECIESCAPE | **INTERNAL ERROR. Ill-formed ECI escape sequence. |
| -23 | ERR_C128CONTENT | **INTERNAL ERROR. Invalid data in the code128 emulation input. |
| -24 | ERR_INVALIDMACROCHAR | **INTERNAL ERROR. Ill-formed macro character substitution sequence in input. |
| -99 | ERR_FATALINTERNAL | **INTERNAL ERROR. Internal API error. |
| -100 | ERR_FONTFATALINTERNAL | **INTERNAL ERROR. Internal conversion error. |
| -101 | ERR_FONTOBJALLOC | **INTERNAL ERROR. A failure occurred during internal buffer memory allocation. |

| Error | Symbolic name | Description |
|-------|-------------------------|--|
| -102 | ERR_FONTROWCOL | <p>Incorrect row/column combination.</p> <p>The ROW parameter can range from 3 to 90. The COLS parameter is converted into an internal code word using this formula:</p> $CW = \text{INT} \left(\left(\text{COLS} * 100 \right) / 425 \right) - 5$ <p>The resultant internal code words can range from 1 to 30. This puts a practical range on the COLS parameter of from 22 to 145.</p> <p>The absolute limit to the size of a bar code is 928 internal code words. All input data and error correction information must be able to be represented by no more than 928 internal code words.</p> |
| -103 | ERR_FONTECC | <p>Incorrect ECC level.</p> <p>Check the ECCLevel parameter value. The valid range is zero (0) through eight (8).</p> |
| -104 | ERR_FONTOUTBUFFTOOSMALL | <p>The output buffer is too small.</p> <p>Check the parameter that defines the size of the output buffer. Make sure it is large enough to represent the size you need.</p> |

If you have multi-page sections, you may get one of these messages, depending on how you set the W parameter in the CreatePDF417Barcode rule.

If you have a multi-page section, you will not get a warning message when the system cannot locate the specified bar code or box if you set the Warning parameter of the CreatePDF417Barcode rule to No. Otherwise, you get the following warning message:

```
DM20402: Warning In CREATEPDF417BARCODE(): The system is unable to
find the specified name for the barcode or box <P417BX1> for image
<AAA> on page <2>.
```


Index

A

AAMVA structure
 retrieving members 15, 23
address for PDF417 Support 5
Adobe Acrobat
 and IDS 14
AFP
 fonts 42
 FXR files 41
American Association of Motor Vehicle Administrators 29
armored hex strings 27
ASCII 29, 32

B

bar codes
 alternate document formats 52
 columns 48
 faxing 52
 overview 3
 producing reliable 52
 rows 50
 size limits 55
 sizing 45
BarCodeName parameter 21, 22, 23, 25, 28
Best Practices Recommendation for the Use of Bar-Codes 4
binary mode 45
bitmap fonts 42

BottomFAPCoord parameter 22, 28
BoxName parameter 25, 28

C

C/C++

- compiler for MVS/ESA 8
- New York application code 29

CCCW32.DLL file 7

certification 12, 14

character sets 44

CheckForRequired parameter 16

COBOL

- interface to application code 29
- output data structure 32
- structure member names 18

code words

- and text columns 46
- defined 45
- size limits 55

columns 48

compulsory insurance laws 3

coordinates 22, 28

CreateNYAAMVADData rule 15

CreatePDF417Barcode rule 21

C-structure member names 19

CUSLIB

- batch processing 13
- merging rules 7, 12
- modifying on MVS 8
- providing customized libraries 11
- replacing with CCCW32.DLL 7

CUSREG member 9

CUSW32.DLL file 7

D

DAL

- functions 26
- variables 19

DALVariableName parameter 27, 28

data definitions 11

data structures

- input 29
- output 32

DDT files 13

Debug_Switches control group 53

digital signatures 29

DLL modules 13

DOCSERV.INI file 25

Docucorp

- support 5

Documaker Workstation 12, 13

DPRLoadImportFile rule 25

driver ID cards 3

E

EBCDIC 29

ECC encoding 45

ECCLevel parameter 21, 23, 25, 28, 54, 55

Enable_Debug_Options option 53

Encrypted 2D Bar Coded ID Card Update II 3, 4

Error Correction (ECC) Encoding 52

error messages 53

F

FAP files 13

faxing

- bar codes 52

- Documaker Workstation 13

Font Manager 41

FontID parameter 21, 22, 23, 25, 28

fonts

- character sets 44

- FXR files 41

- overview 3, 42

formatting input data 40

forms 11

functions 26

FXR files 41

G

GetNYAAMVAVar rule

- defined 23

Guide to the Use of 2-D Bar Codes and Insurance ID

- Cards 4

GVM variables 19

GVMOutputVariable parameter 16, 18

GVMSourceVariable parameter 22, 23

I

IDCardGen Module Changes 4

iDocumaker Workstation 13, 14

IDS

- DLL modules 12

- implementation 14

- rules 25

IIC Security Specifications 4

implementing a solution 11

INI files 13

INI options 18

input data

- formatting rules 40

- mapping 38

INS_Key parameter 16, 17

Insurance Information & Enforcement System (IIES) 3

InsuranceInfo 17

iPPS 13, 14

J

JCL 8

K

Keyword option 26

L

LeftFAPCoord parameter 22, 28

limits

- bar code size 55

LOADPDF member 8

M

mapping data 38

master resource libraries 7, 11

MD5 message digest algorithms 29

messages 53

Metacode

- fonts 42

- FXR files 41

MK_Hard rule 23

MVS

- installation 8

- sample resources 10

N

NAFILE.DAT files 23

New York
 email address 3
 overview of regulations 3
 reference documents 4
 State Insurance Department (NYSID) 5
non-disclosure agreement 15
NYAAMVA control group 16, 18
NYAAMVA.CBL 32
NYID-INSURANCE-INFO 17
NYINSINF.CBL 29

O

OS/390 operating system
 installation 8
output data
 examining 34
 mapping 38
overhead 45

P

P4172FNT message 53
P417DalCreateNYAAMVADData function 27
P417DalCreatePDF417Barcode function 28
P417NYPDF417 rule 25
PassPhrase parameter 16, 17
PCL
 fonts 42
 FXR files 41
PDF417 software
 implementing a solution 11
 installation 5
 on MVS 8
PDF417.LOG file 53
PDF417_2.FXR file 43
PNY01A message 53
PNYLIB option 53
Portable Data File 417 3

porting code 12
PostScript fonts 42
Programmer's Guide to use of IDCardGen Libraries 4
PSYLIB option 53

R

rectangle coordinates 22, 28
registering DAL functions 26
ReqType control group 25
RequiredLevel parameter 27
resources
 for MVS 10
 on Windows 7
responsibilities 11
RightFAPCoord parameter 22, 28
rows 50
rules
 access to 11
 Documaker Server 15
 IDS 25

S

sample resources
 for MVS 10
 for Windows 7
scanners 3
scripts
 registering DAL functions 26
security
 INS_Key parameter 27
 keys 11
signatures 29
Skywire Software
 address 5
 support 5
smoothing 52
support 5

Symbol Technology 21

T

testing the system

 on MVS 9

 on Windows 7

timetable 3

TopFAPCoord parameter 22, 28

tracing 53

Triple DES 29

TrueType fonts 42

TTF fonts 13

U

underscores 48

V

variables 19

W

Windows

 installing under 5

 testing your installation 7

