DieboldNixdorf.com

ProBase Store Migration Guide

For Migrating from ProBase Retail 1.2 or from ProBase POS 2

Table of Contents

Document Information

Document Status

Version:	**1.0**
Date:	**2019/11/29**
Responsible:	Denis Kuniß
Editor(s):	Thomas Heinrich, Fernando Sanabria, Ragnar Seedorf, Oliver Trautmann, Denis Kuniß, Björn Gerhart
Document Status:	Released
Protection Status:	Public

Document History

1.0	Initial release, 06-04-2020

Introduction

ProBase Store is the new platform software product for retail peripherals.

Mainly, it provides a UnifiedPOS [1] library implementation of those peripherals and targets applications based on the programming platforms JavaPOS, OPOS, POS for .NET and CPOS.

Additionally, a Diagnostic & Serviceability Platform application is provided for maintaining and initializing complex cash devices. [2]

The new product is the functional combination of both legacy products, with a new installation structure and enriched by a new Meta configuration concept. ProBase Store version 1 approximately corresponds to the last legacy product releases ProBase POS 2.9.16 and ProBase Retail 1.2.70.

Purpose of This Document

This document describes what have to be considered when migrating either from the Diebold Nixdorf legacy products ProBase POS version 2 or from ProBase Retail version 1.2 to the new product ProBase Store version 1.

The intention is to provide guidance on how to migrate

- when depending on the installation artifacts and their installation structure
- working configurations
- accumulated runtime data

The document is not guiding a new installation of ProBase Store 1 from the scratch. For this please refer to the ProBase Store 1 Installation Guide. [3]

Target Audience

This guide is for integrators and professional service staff which are operating or maintaining a ProBase POS 2 or a ProBase Retail 1.2 installation.

Some information is specific for migrating either from ProBase Retail 1.2 or from ProBase POS 2. In that case the header have a specific tag to allow the other one skip the chapter:

PBR ONLY – for those migrating from ProBase Retail 1.2

PBP ONLY – for those migrating from ProBase POS 2

All chapters without such a tag are for interest for every migration source.

Furthermore, a deeper knowledge of JavaPOS configuration is required. If OPOS, POS for .NET is in use, according knowledge about those platform's configuration concepts is required.

If the Diagnostic & Serviceability Platform application was in use before, knowledge about its configuration is required too.

Professional Service Contact Information

Pre-sales customers should contact their responsible sales / account manager.

Customers who are supported in a professional service project should contact the professional service project team / project manager.

Customers who have been in a professional service project should contact the professional service M&S organization (Maintenance and Service).

Currently, professional service support is limited to Vynamic Checkout customers.

New Product Features and Breaking Changes

Application Specific JavaPOS Configuration PBR ONLY

The ProBase Store product incorporates a new JavaPOS configuration concept already known from ProBase POS 2. This allows an application specific configuration of JavaPOS open names and their configuration properties based on configuration templates provided through the JavaPOS installation.

It also allows to update the underlying ProBase Store installation for an already well configured ProBase Store based application without affecting the productive configuration. For that case, however, the application must be integrated as intended by the ProBase Store configuration mechanism. For details, please refer to chapter “Application Startup Integration” in this document and to the “JavaPOS Configurator” documentation (*<install-dir>\doc\wn-javapos-configurator-userdoc.html*).

Additionally, the concept allows to integrate third party JavaPOS libraries seamlessly if they have been provided as a ProBase Store add-on. Add-ons developed for ProBase POS 2 will work with ProBase Store 1 too.

PBReport for Linux

Formerly PBReport for collecting log files has not been available for Linux at all. Yet, PBReport for Linux is available and provides similar functionality as the one known from Windows. Please see the documentation [4] therefore how to configure and to use it.

No Longer Supported Devices Known from ProBase POS PBP ONLY

Comparing to ProBase POS 2.9 release the following devices are not supported anymore as related service classes has been removed from installation:

CashChanger

- F53,
  - open name: WN_NotesDispenser_JCM_F53
  - service class: com.wn.retail.jpos113.f53.CashChangerF53

MSR

- MSR7816
  - open name:
  - service class: com.wn.retail.jpos113.WNMSR7816’

POSPrinter

- PP01
  - open name: DN_POSPrinter_PP01_USB
  - service class: com.wn.retail.jpos113.posprinter.wnpp0x.WNPP0xPOSPrinterDeviceAdapter
- PP02
  - open name: DN_POSPrinter_PP02_USB

  - service class: com.wn.retail.jpos113.posprinter.wnpp0x.WNPP0xPOSPrinterDeviceAdapter

POSPower

- MPS1086
  - open name: WN_UPS_MPS1086
  - service class: com.wn.retail.jpos113.WNPOSPowerMPS1086

Scanner

- SE3223
  - open name: WN_SE3223_COM
  - service class: com.wn.retail.jpos113.WNScannerSE3223

For the following device no JavaPOS configuration templates are provided anymore, hence they are not available anymore:

CashChanger

- WN_NoteAcceptor_JCMiPro
- WN_NoteAcceptor_JCM_*
- WN_CoinChanger_MCS_*
- WN_CoinChanger_MUX_*
- WN_subCoinAcceptor_MCS_*
- WN_subCoinDispenser_TLQ_*
- WN_CASH_CHANGER_BCR_200

CashDrawer

- WN_CD1_PharmacyPrinter_USB
- WN_CD1_FPTR_EJ210TR – CashDrawer behind the Turkish FiscalPrinter MF-EJ210
- WN_CD1_FPTR_EJ320TR – CashDrawer behind the Turkish FiscalPrinter MF-EJ320
- WN_CD1_FPTR_TH230IT – CashDrawer behind the Italian FiscalPrinter TH230-MF
- WN_CD1_FPTR_EJTHFIT – CashDrawer behind the Italian FiscalPrinter MF-EJTHF
- WN_CD1_FPTR_ND77GR – CashDrawer behind the Greek FiscalPrinter MF-ND77
- WN_CD1_FPTR_EJ210GR – CashDrawer behind the Greek FiscalPrinter MF-EJ210
- WN_CD1_FPTR_EJ320GR – CashDrawer behind the Greek FiscalPrinter MF-EJ320
- WN_CD1_FPTR_ND77RO – CashDrawer behind the Romanian FiscalPrinter MF-ND77
- WN_CD1_FPTR_THFRO – CashDrawer behind the Romanian FiscalPrinter MF-THF

FiscalPrinter

- WN_FPTR_THFRO_COM
- WN_FPTR_EJ320GR_COM
- WN_FPTR_TH230IT_COM
- WN_FPTR_ND77GR_COM
- WN_FPTR_ND77GR_COM
- WN_FPTR_EJ210GR_COM
- WN_FPTR_EJ210GR_COM
- WN_FPTR_EJ210GR_COM
- WN_FPTR_THFIT_COM
- WN_FPTR_ND77RO_COM

LineDisplay

- WN_BA69_SCALE_CONTROLLER
- WN_BA66_USB2 (BA66 still supported, generally)
- WN_BA63_USB2 (BA63 still supported, generally)
- WN_LineDisplay_iPOSplus
- WN_BA63_FPTR_EJ210GR – LineDisplay behind the Greek FiscalPrinter MF-EJ210
- WN_BA66_FPTR_EJ210GR – LineDisplay behind the Greek FiscalPrinter MF-EJ210
- WN_BA63_FPTR_EJ320GR – LineDisplay behind the Greek FiscalPrinter MF-EJ320
- WN_BA66_FPTR_EJ320GR – LineDisplay behind the Greek FiscalPrinter MF-EJ320
- WN_BA63_FPTR_ND77GR – LineDisplay behind the Greek FiscalPrinter MF-ND77
- WN_BA66_FPTR_ND77GR – LineDisplay behind the Greek FiscalPrinter MF-ND77
- WN_BA63_FPTR_ND77RO – LineDisplay behind the Romanian FiscalPrinter MF-ND77
- WN_BA66_FPTR_ND77RO – LineDisplay behind the Romanian FiscalPrinter MF-ND77
- WN_BA63_FPTR_THFRO – LineDisplay behind the Romanian FiscalPrinter MF-THF
- WN_BA66_FPTR_THFRO – LineDisplay behind the Romanian FiscalPrinter MF-THF
- WN_BA63_FPTR_TH230IT – LineDisplay behind the Italian FiscalPrinter TH230-MF
- WN_BA66_FPTR_TH230IT – LineDisplay behind the Romanian FiscalPrinter TH230-MF
- WN_BA63_FPTR_EJTHFIT – LineDisplay behind the Romanian FiscalPrinter MF-EJTHF
- WN_BA66_FPTR_EJTHFIT – LineDisplay behind the Romanian FiscalPrinter MF-EJTHF
- WN_BA63_FPTR_THFIT – LineDisplay behind the Romanian FiscalPrinter MF-THF
- WN_BA66_FPTR_THFIT – LineDisplay behind the Romanian FiscalPrinter MF-THF
- WN_BA63_FPTR_EJ210TR – LineDisplay behind the Turkish FiscalPrinter MF-EJ210
- WN_BA66_FPTR_EJ210TR – LineDisplay behind the Turkish FiscalPrinter MF-EJ210
- WN_BA63_FPTR_EJ320TR – LineDisplay behind the Turkish FiscalPrinter MF-EJ320
- WN_BA66_FPTR_EJ320TR – LineDisplay behind the Turkish FiscalPrinter MF-EJ320

MSR

- WN_MSR7816
- WNMSR_V2X_USB

POSKeyboard

- WN_KB1

POSPrinter

- WN_PharmacyPrinter_USB (only supported on project request)
- WN_TP07C_USB (use the ordinary TP07 open name)

Scale

- WN_SCALE_METTLER_SPIDER-SW (serviceClass = ’com.wn.retail.jpos113.WNScaleMettler’)
- WN_SCALE_METTLER_SPIDER-SW_1_POUND
- WN_SCALE_METTLER_SPIDER-CONFIG
- WN_Scale_Mettler_Toledo_BC15_USB
- WN_SCALE_MAGELLAN_8202, WN_SCALE_MAGELLAN_8502 (serviceClass = ’com.wn.retail.jpos113.WNScalePSC’)

Scanner

- WN_Scanner_Datalogic_RS232_Mode (only supported on project request)
- WN_EL65_COM_DATALOGIC_TD1100
- WN_EL16_COM_anq

- PSC_QS6000_PLUS
- WN_VS1200
- WN_EL36_COM
- WN_VS1200_nr
- WN_SCANNER_USB2_TT (generic table top scanner still available, generally)
- WN_EL53_COM_HONEYWELL_Quantum
- WN_SCANNER_D130_COM
- WN_Scanner_Intermec_ED40
- WN_EL48_COM_HONEYWELL_Voyager
- PSC_8100
  WN_EL16_COM_bnq

Special Electronics

- WN_SpecialElectronicACOSEL_sim
- WN_SELCDL_FOR_MPS
- WN_IOBox_USB

Note: In case any of the devices listed here are still in use in your project, please go in contact with your Diebold Nixdorf representative.

No Longer Supported Devices Known from ProBase Retail PBR ONLY

Comparing to ProBase Retail 1.2.70 release the following devices are not supported anymore as related service classes has been removed from installation:

CashChanger

- F53,
  - open name: WN_NotesDispenser_JCM_F53
  - service class: com.wn.retail.jpos113.f53.CashChangerF53
- iCash15/iCash15E via serviceClass = 'com.wincornixdorf.jpos.changer.icash.cr.CashChangerCR' (service class CashChangerCRA is still provided and is to be used instead)
- iCash15 Simulator (see also RSS-1344)

MSR

- MSR7816
  - open name:
  - service class: com.wn.retail.jpos113.WNMSR7816'

POSPower

- MPS1086
  - open name: WN_UPS_MPS1086
  - service class: com.wn.retail.jpos113.WNPOSPowerMPS1086

Scanner

- SE3223
  - open name: WN_SE3223_COM
  - service class: com.wn.retail.jpos113.WNScannerSE3223

For the following device no JavaPOS configuration templates are provided anymore, hence they are not available anymore:

CashChanger

- WN_NoteAcceptor_JCMiPro
- WN_NoteAcceptor_JCM_*
- WN_CoinChanger_MCS_*
- WN_CoinChanger_MUX_*
- WN_subCoinAcceptor_MCS_*
- WN_subCoinDispenser_TLQ_*
- WN_CASH_CHANGER_BCR_200

CashDrawer

- WN_CD1_PharmacyPrinter_USB

LineDisplay

- WN_BA69_SCALE_CONTROLLER
- WN_BA66_USB2 (BA66 still supported, generally)
- WN_BA63_USB2 (BA63 still supported, generally)
- WN_LineDisplay_iPOSplus

MSR

- WN_MSR7816
- WNMSR_V2X_USB

POSKeyboard

- WN_KB1

POSPrinter

- WN_PharmacyPrinter_USB (only supported on project request)
- WN_TP07C_USB (use the ordinary TP07 open name)

Scale

- WN_SCALE_METTLER_SPIDER-SW via serviceClass = 'com.wn.retail.jpos113.WNScaleMettler' (service class WNScaleMettler1 is still provided and is to be used instead)
- WN_SCALE_METTLER_SPIDER-SW_1_POUND
- WN_SCALE_METTLER_SPIDER-CONFIG
- WN_Scale_Mettler_Toledo_BC15_USB
- WN_SCALE_MAGELLAN_8202, WN_SCALE_MAGELLAN_8502 via serviceClass = 'com.wn.retail.jpos113.WNScalePSC' (service class WNScalePSC1 is still provided and is to be used instead)

Scanner

- WN_Scanner_Datalogic_RS232_Mode (only supported on project request)
- WN_EL65_COM_DATALOGIC_TD1100
- WN_EL16_COM_anq
- PSC_QS6000_PLUS
- WN_VS1200

- WN_EL36_COM
- WN_VS1200_nr
- WN_SCANNER_USB2_TT (generic table top scanner still available, generally)
- WN_EL53_COM_HONEYWELL_Quantum
- WN_SCANNER_D130_COM
- WN_Scanner_Intermec_ED40
- WN_EL48_COM_HONEYWELL_Voyager
- PSC_8100

WN_EL16_COM_bnq

Special Electronics

- WN_SpecialElectronicACOSEL_sim
- WN_SELCDL_FOR_MPS
- WN_IOBox_USB

Note: In case any of the devices listed here are still in use in your project, please go in contact with your Diebold Nixdorf representative.

Supported Java Runtime Environments and Constraints

The generally required Java version for the Java Runtime Environment is Java 8.

Most of the JavaPOS Device Service implementations will still run under Java 6 and 7. However, there are no backward compatibility guarantees given for that.
If a project requires to run on a Java version smaller than 8, it may be validated by testing intensively for a particular Java version and a fixed ProBase Store release. Please note, changes to Java version support for a particular device may be done without further notice. However, we try to document such change properly in the change notes file (*<install-dir>/doc/history/*.changes.txt*) of the particular component.

Newer version of Java above version 8 are not supported yet. However, this may be changed in future versions of the product. Then it will be documented explicitly in the product release notes (<install-dir>/*release-notes.html*).

Please note, the product installers are JRE architecture specific, not really operating system architecture specific! This means, the 64-bit ProBase Store product is only able to run on a 64-bit Java runtime. While, the 32-bit ProBase Store product is only able to run on a 32-bit Java runtime. However, both 32-bit product variants may be installed on a corresponding 64-bit operating system if the 32-bit sub system is installed (this is almost the case for Windows, but not the case for WNLPOS).

Special Restrictions for Cash Devices PBR ONLY

Due to implementation constraints the following cash devices or their special capabilities are not available at following products variants.

ProBase Store Windows 64-bit JRE

- iCash20
- C6010 Secure Channel

ProBase Store Linux 64-bit JRE

- iCash20
- C6010 Secure Channel

Supported Operating Systems and Constraints

Trademarks:

All company names and trademarks mentioned in the list are the property of their respective owners.

ProBase Store is released for the following operating systems:

- Microsoft Windows® 7™ Professional x86
- Microsoft Windows® 7™ Professional x64
- Microsoft Windows® Embedded POSReady 7™ x86
- Microsoft Windows® Embedded POSReady 7™ x64
- Microsoft Windows® Embedded 8.1™ Industry Professional x86
- Microsoft Windows® Embedded 8.1™ Industry Professional x64
- Microsoft Windows® 10™ IoT Enterprise x64
- Diebold Nixdorf WNLPOS 3
- Diebold Nixdorf WNLPOS 4

The following operating systems supported by previous versions of Probase POS and Probase Retail are no longer supported by Probase Store:

- Microsoft Windows® Embedded POSReady 2009 x86
- Microsoft Windows® XP Professional x86
- Diebold Nixdorf WNLPOS 2

Additionally, see note at chapter “Supported Java Runtime Environments and Constraints”.

Changes to Deployment Artifacts

The ProBase Store product consists of versioned components which are aggregated and provided through a single product release installer. This concept follows those which was already applied to ProBase POS 2.

The components and their versions may be investigated through their change notes files *(<install-dir>/doc/history/*.changes.txt*) and version files (.

Drivers

The JavaPOS Device Service implementations are relying on Windows system drivers. Normally, those are coming preinstalled on Diebold Nixdorf maintained Beetle system. They are not part of the ProBase Store product distribution (as it was the case for ProBase Retail 1.2). However, all required drivers, their versions and which particular devices required them are documented in the release notes (*<install-dir>/release-notes.html*), chapter “Driver Dependencies”. There is also stated how to obtain them from the Diebold Nixdorf Internet web site.

For Linux no particular driver is required.

Loadutil PBR ONLY

The Loadutil installer is now provided separately and it does not follow the complementary installation component concept. Instead, a single, standalone installer includes all Loadutil functionality. There is one installer per architecture and the following is the installation path:

Windows 32 bits: *C:\Program Files (x86)\Diebold Nixdorf\Loadutil*
Windows 64 bits: *C:\Program Files\Diebold Nixdorf\Loadutil*
Linux: */opt/dn/loadutil*

Loadutil now brings its own embedded JRE and which gets installed into the *jre* sub-directory under the installation path. No support for alternative JREs is included, it runs on its own JRE only.

Loadutil log files are located under the following paths:

Windows: *%ProgramData%\Diebold Nixdorf\RSS\loadutil*
Linux: */var/log/dn/loadutil*

Changes to the Installation Structure

The installation structure of ProBase POS is identical to ProBase Store.

On the other hand the installation structure of ProBase Retail is completely different.

Linux

The installation structure is according to Linux Standard Base [5], so the installation location of the artefacts depends on their purpose:

Directory Structure

/opt/wn/javapos/ – main installation path of Diebold Nixdorf JavaPOS artefacts

/opt/wn/javapos/doc/ – documentation

/opt/wn/javapos/lib/ – java libraries that will be put into CLASSPATH

/opt/wn/javapos/bin/ – native binaries, scripts

/opt/wn/javapos/xml/ – JavaPOS Configuration templates

/opt/dn/diagserv/ – Diagserv/TSOP installation folder

/opt/dn/pbreport/ – PBReport installation folder

Runtime and Logging Data

/var/log/dn/ – log files

/var/log/wn/javapos/ – log files (wn_javapos_log_home)

/etc/opt/dn/jdd/ – JDD log files and persistence

/var/opt/wn/ – persistence / program data (wn_javapos_data_home)

Configuration Files

/etc/opt/wn/javapos/config/ – Java configuration files that must be put into CLASSPATH

/etc/opt/javapos/ – JavaPOS Configurator generated JavaPOS configuration(s)

/etc/opt/dn/pbreport/ – PBReport rules

Windows

Depending on chosen installer (ProBase Store 32 bit vs. ProBase Store 64 bit) the main installation directory is either

C:\Program Files\javapos\ProBase-Store

or

C:\Program Files (x86)\javapos\ProBase-Store

In following section “C:\Program Files\javapos\ProBase-Store” stands for both options.

Directory Structure

C:\Program Files\javapos\ProBase-Store\ – main installation folder

C:\Program Files\javapos\ProBase-Store\bin\ – native libraries, scripts

C:\Program Files\javapos\ProBase-Store\config\ – Java configuration files that must be put into CLASSPATH

C:\Program Files\javapos\ProBase-Store\doc\ – documentation

C:\Program Files\javapos\ProBase-Store\lib\ – java libraries that will be put into CLASSPATH

C:\Program Files\javapos\ProBase-Store\xml\ – JavaPOS Configuration templates

C:\Program Files\Diebold Nixdorf\ – main installation folder for non-JavaPOS tools and apps

C:\Program Files\Diebold Nixdorf\Cm3srTeachtool\ – CM3SR (C1030) Teachtool being aggregated by TSOP

C:\Program Files\Diebold Nixdorf\Diagserv\ – D&S/TSOP service application

C:\Program Files\Diebold Nixdorf\PBReport\ – PBReport tool to collect logs and system information

Runtime and Logging Data

C:\ProgramData\javapos\ – JavaPOS Configurator output; JavaPOS logs

C:\ProgramData\Diebold Nixdorf\JDD\ – JDD persistence and runtime data

C:\ProgramData\Diebold Nixdorf\RSS\ – Retail System Software persistence, runtime data, and logs

Configuration Files

C:\Program Files\javapos\ProBase-Store\config\ – configuration files

Changes to Configuration Concepts

Application Startup Integration PBR ONLY

As mentioned in chapter “Application Specific JavaPOS Configuration” a specific JavaPOS configuration mechanism has been incorporated into ProBase Store which was already in place at ProBase POS 2.

To get a proper startup environment prepared for an ProBase Retail 1.2 integrated application which wants to migrate to ProBase Store, each call to the shell script *C:\Retail\Software\javapos\setjavaposvars.bat* respectively */usr/local/javapos/setjavaposvars.sh* at application start scripts have to be substituted by calls to shell script *<%ProgramData%>\javapos\setenv.bat* and */etc/opt/javapos/setenv.sh* respectively.
All environment variables previously available through *setjavaposvars* shell scripts are also available through the new *setenv* shell scripts.

New JavaPOS Configuration File Naming Schema

The main goal for the configuration file naming schema is to make it for the integrator easier to find a particular device configuration in the set of installed JavaPOS XML configuration files. Therefore, names and abbreviations are used which are either well known for the integrator, like UnifiedPOS (UPOS) category names, or easy to figure out, e.g. HW portfolio names.

Generally, the product organizes the device open names into configuration files according to their UnifiedPOS category. So the general configuration file naming schema is

```
dn-javapos-<upos category>.xml
```

For internal reasons, some devices needs to be contained in separate files. Therefore an additional indicator is used which almost is associated with the HW device portfolio name or portfolio category. This indicator is added as suffix to the category name:

```
dn-javapos-<upos category>-<hw-device-category>.xml
```

For hydra devices, like a POS printer with line displays or cash drawers connected behind, the open name configurations are split into separate files. The non-main devices (those not connected to the host) are defined in configuration files with their category name separated by dot:

```
dn-javapos-<upos category main device>.<upos category non-main device>.xml
```

So, generally all Diebold Nixdorf JavaPOS configuration files starts with the prefix "dn-javapos-" followed by the UnifiedPOS category name for which it contains open name configurations.

Under circumstances, the “dn-javapos-“ is not followed by the category name, but by a specific name associated by the component it is provided by. It typically contains open name configurations of several device categories which are belonging together for technical reasons (e.g. forming together a compound device).

Examples

dn-javapos-cashdrawer.xml – contains CashDrawer open name configurations.

dn-javapos-posprinter-th250.xml – contains POSPrinter open name configurations for the TH250 printer type.

dn-javapos-posprinter.linedisplay.xml – contains LineDisplay open name configurations for devices connected to a POS printer.

dn-javapos-portalscanner.xml – contains all devices to operate an integrated portal scanner devices.

New Open Name Schema

The main goal for open names is to make them easy to be associated with a particular hardware device for the integrator. Therefore, names and abbreviations are used which are either well known for the integrator, like UnifiedPOS (UPOS) category names, or easy to figure out, e.g. HW portfolio names.

Furthermore, a portfolio device may have different HW variants which are typically reflected by a particular configuration variants in an open name configuration (e.g., system port configurations for RS232 or USB).

Additionally, there may be HW identification variants in case the same HW device type is used several times at one system (e.g., 2 USB based TP07; or same typed USB scanner, one handheld, one table top).

So generally, the following open name naming schema is applied:

```
DN_<upos-category>_<portfolio-or-generic-name>_<variant1>_<variant2>…
```

The prefix "DN_" is fixed for recognizing Diebold Nixdorf as the HW provider of the device. It is followed by the UnifiedPOS category the device belongs to (programmatic name is used).

Under circumstance, the generic name is omitted if the category name is equal to the generic name.

For devices not connected directly to the host (non-main devices), like a line display connected behind a POS printer, a special naming schema is used to let recognize the special situation:

```
DN_<non-main-upos-category>_<portfolio-or-generic-name>_<variant>…_<main-device-open-name>
```

Examples

`DN_POSPrinter_TH230_USB` – the open name for the Diebold Nixdorf POS printer type TH230 with USB interface

`DN_POSPrinter_TP07_USB_DEV1` - the open name for the Diebold Nixdorf POS printer TP07 which is identified by the device serial number written to a configuration property of this open name configuration, associated in the system as "DEV1" (because there exists also a "DEV2" which is slightly different configured then).

`DN_LineDisplay_BA66_POSPrinter_TH230_USB` - is the open name for the Diebold Nixdorf's line display type BA66 connected to the USB based POS printer TH230 configured as open name `DN_POSPrinter_TH230_USB`

New Approach to Cash Devices Configuration

As mentioned in chapter “Application Specific JavaPOS Configuration” a specific JavaPOS configuration has to be created using the JavaPOS Configurator which is based on configuration templates for the specific devices or device families.

In ProBase Retail 1.2 a configuration example was provided for each kind of a device or device family. The new approach replaces multiple old configuration examples with one single template configuration per device family represented by a new open name given, according to the new name schema as described in chapter “New Open Name Schema”.

Due to the replacement of multiple examples by a single template, this template cannot reflect HW device specific properties like the specific roller configuration. These properties are missing in the templates and has to be added using the new configuration mechanism via a dedicated *javapos.config.properties* file. For details, please refer to the “JavaPOS Configurator” documentation (*<install-dir>\doc\wn-javapos-configurator-userdoc.html*).

Examples

The specific number of cash units at a concrete C6020 device is not reflected by the *C6010* open name configuration template.

Also, the different variants of a C6010 device (with Cassette or E2E-Roller; with/without Secure Channel; C6010 stand-alone or C6010 as part of a SCO) are not reflected by the *C6010* configuration template.

The mapping of old configuration examples to new configuration templates can be found in chapter “Cash Devices Configuration Using Templates” (see Table 1: Configuration of Cash Devices – Mapping of examples to templates). Also the new mechanism with an application dedicated *javapos.config.properties* file is described in this chapter.

Field Migration Hints PBR ONLY

Cash Counter Migration C1030

Counter information of C1030 is stored on file system and there is no automatic counter migration implemented.

That means after fresh ProBase Store installation the counter are reset to zero no matter what the physical level are.

So either

A) after ProBase Store installation a full stock check is done by emptying the device completely and paying in all coins afterwards, or

B) counter information of ProBase Retail 1.2.* is manually migrated before ProBase Store is used the first time. To do so process the following copy instructions:

- copy C:\Retail\Software\cindrv\jddConfigFiles\data.coin_engine\HopperLevel.properties to C:\ProgramData\Diebold Nixdorf\JDD\cm3srCoinEngineHome\data.coin_engine\HopperLevel.properties
- copy C:\Retail\Software\cindrv\jddConfigFiles\data.coin_engine\backup\HopperLevel.properties to C:\ProgramData\Diebold Nixdorf\JDD\cm3srCoinEngineHome\data.coin_engine\backup\HopperLevel.properties
- copy C:\Retail\Software\cindrv\jddConfigFiles\data.coin_engine\OverFlowBox.properties to C:\ProgramData\Diebold Nixdorf\JDD\cm3srCoinEngineHome\data.coin_engine\OverFlowBox.properties
- copy C:\Retail\Software\cindrv\jddConfigFiles\data.coin_engine\backup\OverFlowBox.properties to C:\ProgramData\Diebold Nixdorf\JDD\cm3srCoinEngineHome\data.coin_engine\backup\OverFlowBox.properties

Note: ProBase Store Linux does not support C1030 at all.

Security Configuration C1030

In contrast to former product ProBase Retail no such JRE options must be set:

```
-Djava.security.manager
-Djava.security.policy="C:\Program Files\javapos\ProBase-Store\config\java.policy"
```

The reason was the usage of a SecurityManager demanded by CM3SR CoinEngine layer.

With ProBase Store it is still demanded, but required policies are applied programmatically now just before the CoinEngine gets opened.

Note: ProBase Store Linux does not support C1030 at all.

Cash Devices Configuration Using Templates

As described in chapter “New Approach to Cash Devices Configuration” ProBase Store no longer provides example configurations for the application (here, TPCash).

The new configuration mechanism is based on a *javapos.config.properties* file which adds device or application specific configuration parameters to an application dedicated open name configuration derived from a configuration template for a particular device family.

The following table shows the mapping from open name configurations (XML file and open name) provided by Probase Retail 1.2 to new open name template configurations (XML file and open name) provided by ProBase Store 1.

ProBase Retail 1.2	ProBase Store 1.0
wincor.jpos113.cineo.xml CashChanger_apats_EUR CashChanger_apats_BEE CashChanger_apats_XTS CashChanger_apats_safebag_XTS configICash.xml apats-1 apats-2	dn-javapos-cashchanger-rm3-ats.xml DN_CashChanger_RM3_ATS
wincor.jpos113.cineo.xml CashChanger_appos_EUR CashChanger_appos_EUR_without_SEL CashChanger_appos_BEE CashChanger_appos_XTS CashChanger_appossc_EUR CashChanger_appossc_BEE CashChanger_appossc_XTS configICash.xml appos-1 appos-2 appos-3-towerline	dn-javapos-cashchanger-rm3-pos.xml DN_CashChanger_RM3_POS
wincor.jpos113.cineo.xml CashChanger_C1020	
wincor.jpos113.cineo.xml CashChanger_C1030 configICash.xml c1030-1	dn-javapos-cashchanger-c1030.xml DN_CashChanger_C1030
wincor.jpos113.icashxx.xml CashChanger_iCash10 CashChanger_iCash30 CashChanger_iCash50 CashChanger_iCash100 configICash.xml ic10-1, ic10-2	dn-javapos-cashchanger-rz.xml DN_CashChanger_iCash10 DN_CashChanger_iCash30

ic30-1 ic50-1, ic50-2 ic100-1, ic100-2	
wincor.jpos113.icashxx.xml configICash.xml ic15-1, ic15-2, …, ic15-10 ic15e-1, ic15e-2, …, ic15e-10 ic15CO-1	dn-javapos-cashchanger-cra.xml DN_CashChanger_CRA
wincor.jpos113.icashxx.xml CashChanger_iCash15_CR	No longer supported device service implementation as described in “No Longer Supported Devices Known from ProBase Retail”. Alternate implementation could be used: dn-javapos-cashchanger-cra.xml DN_CashChanger_CRA
wincor.jpos113.icashxx.xml CashChanger_iCash20 configICash.xml ic20-1	dn-javapos-cashchanger-icash20.xml DN_CashChanger_iCash20

Table 1: Configuration of Cash Devices – Mapping of examples to templates

To support the migration from ProBase Retail 1.2 to ProBase Store 1 an application specific example configuration is provided for TPCash. The file *javapos.config.properties* in directory *%PROGRAMDATA%\javapos\tpcash* and */etc/opt/javapos/tpcash/javapos.config.properties*, respectively, provides the most important configuration parameters of the devices used by TPCash.

At the beginning of each line a “`#`” sign is used to deactivate the configuration of the line. This sign has to be removed to activate a required configuration. The file *javapos.config.properties* is a Java properties file and “#” is the comment indicator for such file types.

The devices used by a TPCash installation has to be activated using the line starting with “`#jpos.names=`”. It is mandatory to activate this line (by removal of “#”). The list of devices (open names) has to be modified according to devices in use for a particular TPCash installation.

Each device (open name) activates a set of parameters in the sections later in the file. For example, the setting “`jpos.names=C6010,C1010`” activates the configuration for a C6010 and a C1010 device.

For each activated device at least the parameter “`#jpos.name.<open_name>=`” has to be commented in. For C6010 and C1010 the following parameters has to be activated:

```
jpos.name.C6010=DN_CashChanger_RM3_POS
jpos.name.C1010=DN_CashChanger_C1010
```

If device specific parameters has to be changed or added, the relevant lines has to be enabled as follows.

For a C6010 with 4 cash units (1 E2E roller and 3 standard rollers) the following parameters should be commented in:

```
# fixed drum 1 - configured as deposit unit (E2E)
jposentry.C6010.roller1.role=DEPOSIT
jposentry.C6010.roller1.allCategory4=true
jposentry.C6010.roller1.allCategory3=true
jposentry.C6010.roller1.allCategory2=true
jposentry.C6010.roller1.allUnfit=true
jposentry.C6010.roller1.allMaculated=true
jposentry.C6010.roller1.individualCashTypes=
jposentry.C6010.roller1.capacity=300
# fixed drum 1 - configured as recycling unit (only in systems with cassette 1!)
jposentry.C6010.roller1.role=RECYCLE
jposentry.C6010.roller1.allCategory4=false
jposentry.C6010.roller1.allCategory3=false
jposentry.C6010.roller1.allCategory2=false
jposentry.C6010.roller1.allUnfit=false
jposentry.C6010.roller1.allMaculated=false
jposentry.C6010.roller1.individualCashTypes=EUR:20:0
jposentry.C6010.roller1.capacity=300
# fixed drum 2 - configured as recycling unit
jposentry.C6010.roller2.role=RECYCLE
jposentry.C6010.roller2.allCategory4=false
jposentry.C6010.roller2.allCategory3=false
jposentry.C6010.roller2.allCategory2=false
jposentry.C6010.roller2.allUnfit=false
jposentry.C6010.roller2.allMaculated=false
jposentry.C6010.roller2.individualCashTypes=EUR:5:0
jposentry.C6010.roller2.capacity=300
# fixed drum 3 - configured as recycling unit
jposentry.C6010.roller3.role=RECYCLE
jposentry.C6010.roller3.allCategory4=false
jposentry.C6010.roller3.allCategory3=false
jposentry.C6010.roller3.allCategory2=false
jposentry.C6010.roller3.allUnfit=false
jposentry.C6010.roller3.allMaculated=false
jposentry.C6010.roller3.individualCashTypes=EUR:10:0
jposentry.C6010.roller3.capacity=300
```

After having activated these parameters, installation specific adaptations have to be done. For example, the used currency or denomination of each cash unit should be adapted.

Additional properties may be added according to the device specific documentation.

For the C1010 no additional configuration parameters are provided within the TPCash specific *javapos.config.properties* file. If needed, additional properties could be added according to the device specific documentation.

T/SOP Configuration

According to the changes described in the chapters before the new open names configured through the new JavaPOS configuration mechanism partially have to be used in the D&S-T/SOP configuration file *config.properties* too, for those T/SOP device configurations which require the JavaPOS open name of the T/SOP controlled device to be the suffix of the T/SOP device name.

The provided *config_*.properties* examples coming preinstalled with a ProBase Store 1 (at *<diagserv-install-dir>/configuration*) use T/SOP device names associated with JavaPOS template open names (using them as suffix for the affected T/SOP device configurations). A specific open name configuration at an application dedicated *javapos.config.properties* file is not considered automatically and has to be adjusted by manually by adapting the JavaPOS open name suffix used for the dedicated application (planned to be improved in future versions to be configured automatically).

Upgrading Procedures

Upgrading from ProBase POS 2 PBP ONLY

ProBase Store's installation and configuration concepts has been derived from ProBase POS 2. So, the migration from ProBase POS 2 is quite easy and contains the following steps. However, there was a small change at the Linux version therefore for Linux it is slightly more to do.

Windows

1. Backup your configuration files at *%ProgramFiles%/javapos/*/javapos.config.properties*.
   Please note the asterisk – it could be more than one file!
2. Uninstall ProBase POS – by selecting the according ProBase POS Windows Start menu entry.
3. Install ProBase Store – by executing the installer executable.
4. Copy the backed up *javapos.configuration.properties* files to their original location (maybe overriding the newly installed ones).
5. Adapt the template open names in the original *javapos.config.properties* files to the new ones. The mapping list of old to new template open names can be obtained from the documentation "Supported Devices" in the ProBase Store Windows start menu under "Devices", a HTML browser with a table of supported open name templates will open. It contains a column for the old templates open name templates.
6. Run the JavaPOS Configurator from the ProBase Store's Windows start menu (entry Configuration->Generate JavaPOS configuration) and check your application configuration at the ProBase Store's Windows start menu entry "Configured Devices", a HTML browser with a table of configured open names will open.

Linux

1. Backup your configuration files at */etc/opt/*/javapos.config.properties*.
   Please note the asterisk – it could be more than one file!
2. Uninstall ProBase POS: `rpm -e wn-probase-pos.`
3. Install ProBase Store: `rpm -i dn-probase-store-1.0.*.rpm`
4. Copy the backed up *javapos.configuration.properties* files to */etc/opt/javapos/*/ javapos.config.properties*.
   Please note, this is not their original location! The files and their containing directories have been moved one level up into the *javapos* directory.
5. Adapt the template open names in the original *javapos.config.properties* files to the new ones. The mapping list of old to new template open names can be obtained from the documentation "Supported Devices" in the ProBase Store Linux application menu, a HTML browser with a table of supported open name templates will open. It contains a column for the old open name templates.
6. Run the JavaPOS Configurator `/opt/wn/javapos/bin/config_javapos_startup.sh` (or select the Gnome menu entry "Generate JavaPOS configuration") and check your application configuration at the ProBase Store's Linux application menu entry "Configured Devices", a HTML browser with a table of configured open names will open.
7. Remove the original configuration files */etc/opt/*/javapos.config.properties* you have backuped before in point 1. They are not in use anymore.

Upgrading from ProBase Retail 1.2 PBR ONLY

ProBase Retail's installation and configuration concepts are different to ProBase Store one's, so it is recommended to uninstall ProBase Retail before ProBase Store gets installed. It is not intended, and not supported to run ProBase Retail and ProBase Store in parallel. Proceed the following steps to upgrade from ProBase Retail to ProBase Store:

Windows

1. Backup logs, configuration files, and other files of your interest (especially, see the chapter "Cash Counter Migration C1030" for cash counter backup)
2. Uninstall ProBase Retail
3. Delete folder that may not have been deleted during uninstallation because they contained log files, modified configuration files, or some additional files:
   a. C:\Retail\Software\autoconf
   b. C:\Retail\Software\cindrv
   c. C:\Retail\Software\cineofw
   d. C:\Retail\Software\diagserv
   e. C:\Retail\Software\javacim
   f. C:\Retail\Software\javapos
   g. C:\Retail\Software\probase_retail
   h. C:\Retail\Software\probase_retail_firmware
   i. C:\Retail\Software\Tools\LoadUtil2
4. Delete Registry keys that may have been persisted:
   a. HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Wincor Nixdorf\Autoconf
   b. HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Wincor Nixdorf\CSC-W32
   c. HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Wincor Nixdorf\DIAGSERV
   d. HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Wincor Nixdorf\JavaCIM
   e. HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Wincor Nixdorf\JavaPOS
   f. HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Wincor Nixdorf\ProBase Retail
   g. HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Wincor Nixdorf\ProBaseRetail
   h. HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Wincor Nixdorf\Tools\Loadutil2
   i. HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Wincor Nixdorf\WNreport
   j. HKEY_LOCAL_MACHINE\SOFTWARE\Wincor Nixdorf\Autoconf
   k. HKEY_LOCAL_MACHINE\SOFTWARE\Wincor Nixdorf\CSC-W32
   l. HKEY_LOCAL_MACHINE\SOFTWARE\Wincor Nixdorf\DIAGSERV
   m. HKEY_LOCAL_MACHINE\SOFTWARE\Wincor Nixdorf\JavaCIM
   n. HKEY_LOCAL_MACHINE\SOFTWARE\Wincor Nixdorf\JavaPOS
   o. HKEY_LOCAL_MACHINE\SOFTWARE\Wincor Nixdorf\ProBase Retail
   p. HKEY_LOCAL_MACHINE\SOFTWARE\Wincor Nixdorf\ProBaseRetail
   q. HKEY_LOCAL_MACHINE\SOFTWARE\Wincor Nixdorf\Tools\Loadutil2
   r. HKEY_LOCAL_MACHINE\SOFTWARE\Wincor Nixdorf\WNreport
5. Install ProBase Store
6. Adapt/Create your application *javapos.config.properties* file so the JavaPOS Configurator creates your target JavaPOS configuration. The template open names has been changed, the mapping list of old to new template open names can be obtained from the documentation "Supported Devices" in the ProBase Store Windows start menu under "Devices", a HTML browser with a table of supported open name templates will open. It contains a column for the old templates open name templates.

7. Run the JavaPOS Configurator from the ProBase Store's Windows start menu and check your application configuration at the ProBase Store's Windows start menu entry "Configured Devices", a HTML browser with a table of configured open names will open
8. Configure Diagserv by editing *C:\Program Files\Diebold Nixdorf\Diagserv\configuration\config.properties*.

Linux

1. Backup logs, configuration files, and other files of your interest
2. Uninstall ProBase Retail: `/usr/local/probase_retail/uninstall-ProBaseRetail.sh`
3. Delete folder that may not have been deleted during uninstallation because they contained log files, modified configuration files, or some additional files:
   a. /etc/opt/dn/pbreport/rules.d
   b. /etc/opt/dn/pbreport
   c. /etc/opt/wn/pbreport/rules.d
   d. /etc/opt/wn/pbreport
   e. /etc/opt/wn/javapos
   f. /etc/opt/javapos
   g. /etc/udev/rules.d/51-cineo.rules
   h. /etc/udev/rules.d/52-mei.rules
   i. /etc/udev/rules.d/80-crypta-mount.rules
   j. /etc/udev/rules.d/81-crypta-usb-access.rules
   k. /etc/udev/rules.d/99-wn-javapos-th250.rules
   l. /etc/init.d/wn-startup
   m. /var/log/dn/loadutil
   n. /var/log/wn/javapos
   o. /var/opt/wn/javapos/lock/wn_wndev_lock
   p. /var/opt/dn/javapos/lock/wn_wndev_lock
   q. /opt/wn/javapos
   r. /opt/dn/javapos
   s. /opt/dn/diagserv
   t. /opt/wn/diagserv
   u. /opt/wn/autoconf
   v. /opt/dn/autoconf
   w. /usr/local/javapos
   x. /usr/local/javacim
   y. /opt/wn/tools/loadutil2
4. Install ProBase Store: `rpm -i dn-probase-store-1.0.*.rpm`
5. Adapt/Create your application *javapos.config.properties* file so the JavaPOS Configurator creates your target JavaPOS configuration. The mapping list of old to new template open names can be obtained from the documentation "Supported Devices" in the ProBase Store Linux application menu, a HTML browser with a table of supported open name templates will open. It contains a column for the old open name templates.
6. Run the JavaPOS Configurator `/opt/wn/javapos/bin/config_javapos_startup.sh` (or select the Gnome menu entry "Generate JavaPOS configuration") and check your application configuration at the ProBase Store's Linux application menu entry "Configured Devices", a HTML browser with a table of configured open names will open.

References

[1] ARTS-NRF, UnifiedPOS Retail Peripheral Architecture; Version 1.13; For Implementation of Point Of Service Peripherals, July 15, 2009.

[2] Diebold Nixdorf, Diagnostics and Serviceability Platform - Integration Manual, version 1620.

[3] Diebold Nixdorf, ProBase Store 1 Installation Guide, 2019.

[4] Diebold Nixdorf, "pbreport Manual".*Probase Store Installation, file:///opt/dn/pbreport/pbreport.html.*

[5] Wikipedia.org, "Linux Standard Base".*Wikipedia.*