Omni-ID, Inc

**MIDDLEWARE INSTALLATION GUIDE**

*Document Revision: 1.0*

*Date of last revision: September 17, 2013*

*Omni-ID, Inc.*

*1200 Ridgeway Avenue*

*Rochester, NY*

[www.Omni-ID.com](http://www.Omni-ID.com)

*Author: Omni-ID Software Development Team*



# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Revision** | **Author** | **Notes** |
| 9/17/13 | 1.0 | Huzyk | Initial document |

## Introduction

The Omni-ID Middleware solution consists of multiple software packages, database tables, and runtime libraries. This document is the master set of instructions for the solution. It will provide a sequence of installation, installation and configuration instructions, and references to other documents that have installation and configuration instructions.

## Middleware Installation

The Middleware solution should be installed in the following order:

1. Turn Windows UAC (User Account Control) to off or its lowest level.
2. Bonjour for Windows
3. Omni Binaries (see section 4 below)
4. CALC Manager (see *CALC Manager installation guide.docx*)
5. Additional database tables (see section 3 below)
6. Import Connection String Utility (see section 5 below)
7. Image Generator Service (see section 6 below)
8. System Scheduler Service (see section 7 below)
9. Omni Impinj Reader application (see section 8 below)
10. Middleware libraries (see *Omni NET API 1\_0.docx,* be sure to create the QueueNames.txt file mentioned in section 5.1 of that document)

## Additional Database Tables

Execute the supplied scripts in order:

1. OmniAppCreate.sql
2. OmniAppInsert.sql

## Omni Binaries (Log Service and Impinj Reader)

## Installation

At this time, there is no installer for the Omni Log Service and Impinj readers, so the files must manually be copied from the \binaries folder to the “C:\Program Files (x86)\OMNI-ID\bin\” folder.

## Import Connection String Utility

## Summery

The Import Connection String utility is used to import the database connection string into a datastore for use by some of the other software components.

## Installation

Install the OmniImportConnectionStringSetup.msi file.

## Setup

Run the OmniImportConnectionString app from the start menu. Type your database connection string into the “New Connection string” text box”, and press the Import button.

## Image Generator Service

## Summery

The Image Generator Service is an automatic start windows service that the CALC Manager uses to create the pages for the CALCs.

## Installation

Install the ImageGenSetup.msi file.



## System Scheduler Service

## Summary

The System Scheduler Service provides a mechanism to start applications according to different schedule types (always once, interval, always running, etc). It gets it’s configuration from the CALCMan database.

## Installation

Install the OmniSysSchedServiceInstaller.msi file.

## Setup

The system scheduler service is configured by the scripts in section 3 above. Additional information on these configuration tables can be found in Appendix A.

## Omni Impinj Reader Application

## Installation

The Omni Impinj Reader is currently installed in section 4 above.

## Setup

TBD

## Appendix A – System Scheduler Database Tables

# OmniApplication\_\* Database Tables description

## Purpose:

### These database tables provide a very flexible means to store and retrieve configuration data for applications that are started and monitored by the Omni System Scheduler service.

### The Omni System Scheduler service is a Windows Service that periodically scans the OmniApplication\_ScheduledApps table. Each row in this table represents one application that the Omni System Scheduler is to start, and possibly monitor and if found not to be running, to restart.

## List of Tables:

### OmniApplication\_Application

### OmniApplication\_Parameter

### OmniApplication\_ParameterCategory

### OmniApplication\_ParameterType

### OmniApplication\_ScheduledApps

### OmniApplication\_ScheduleType

### OmniApplication\_YesNo

## Detailed explanation of each table:

### OmniApplication\_Application

#### Each Application that requires storage and retrieval of configuration parameters has one row in this table.

#### The columns in this table are:

|  |  |
| --- | --- |
| **Column Name** | **Description** |
| ID | Autoincrementing integer |
| Name | Name of application – unique constraint |
| GUID | Unique constraint, meant to store ‘stringified’ GUIDs |

### OmniApplication\_ParameterCategory

#### Each row encapsulates a definition of a named grouping of configuration parameters for a particular application.

#### The columns in this table are:

|  |  |
| --- | --- |
| **Column Name** | **Description** |
| ID | autoincrementing integer |
| Name | Name of the parameter category – unique per application |
| Description | optional description string of the parameter category |
| FK\_ID\_OmniApplication\_Application | Foreign key reference to a row in the OmniApplication\_Application table ; designates to which Application this parameter category applies. |

### OmniApplication\_ParameterType

#### Used to define which data types of parameters are supported.

#### Currently, only the ‘string’ type is supported.

#### Referenced by rows in the OmniApplication\_Parameter table, so signify what the type of the parameter value is.

#### The columns in this table are:

|  |  |
| --- | --- |
| **Column Name** | **Description** |
| ID | Autoincrementing integer |
| Name | Name of the parameter type |
| Description | Description of the parameter type |
| ValidationRegularExpression | Optional regular expression to validate user input strings as being of the correct format for this type. |

### OmniApplication\_Parameter

#### Each row stores the name, type, value, parameter category, and application of a particular configuration parameter

#### The columns in this table are:

|  |  |
| --- | --- |
| **Column Name** | **Description** |
| ID | Autoincrementing integer |
| FK\_ID\_OmniApplication\_Application | References a row in the OmniApplication\_Application table, to designate to which application this parameter applies. |
| FK\_ID\_OmniApplication\_ParameterType | References a row in the OmniApplication\_ParameterType table, to give the parameter type. |
| FK\_ID\_OmniApplication\_ParameterCategory | References a row in the OmniApplication\_ParameterCategory table, to give which parameter category this parameter belongs to. |
| Name | Gives the name of the parameter |
| Value | Stores a string representation of the parameter’s value. |

### OmniApplication\_ScheduleType

#### Each row represents a scheduling algorithm that may be applied to a particular application managed by the OmniSystemScheduler.

#### Four Scheduling Types are supported, each type represented by a single row in this table.

#### The scheduling types are:

##### Always Running – OmniSystemScheduler is to start the application if it is not running, and periodically verify the application is running. If the application is found not to be running, then the OmniSystemScheduler will re-start the application.

##### Daily At Time – OmniSystemScheduler will run the specified application once per day at a specified time.

##### Interval – OmniSystemScheduler will run the specified application periodically at a specified internal, such as every 1 hour.

##### Once – OmniSystemScheduler will run the specified application only once, starting at the specified time.

### OmniApplication\_YesNo

#### A simple ‘enumeration table’ to encode the values of “Yes” and “No”

#### Was necessary to support multiple data storage schemes / database types by the OmniSystemSchedulerSystem.

### OmniApplication\_ScheduledApps

#### This is table that the OmniSystemScheduler Windows Service will periodically scan to determine which applications in its database to start, or monitor and possibly re-start.

#### Each row stores the necessary information to support the ability to start a Windows application, and store the information necessary to support the four available schedule types.

#### The columns in this table are:

|  |  |
| --- | --- |
| **Column Name** | **Description** |
| ID | Autoincrementing integer |
| AppName | Name of the application – unique constraint. |
| FullyQualifiedExePath | Contains the fully qualified path of the Windows executable to start. |
| WorkingDirectoryPath | Optionally provide the working directory the launched application should use. |
| CommandLine | Optionally provide command line parameters for the application being launched. |
| HideGUI | YesNo type (1 = Yes, 2 = No) |
| AppGUID | Stores stringified GUID to uniquely identify this scheduling entry. |
| dtLastRun | datetime – stores the timestamp the application was last run |
| dtNextRun | datetime – stores the timestamp the application should next be run |
| RunIntervalMinutes | integer – stores the interval at which the application should be periodically started. |
| FK\_ID\_ScheduleType | References a row in the OmniApplication\_ScheduleType table to give the schedule type , such as “Always Running” for the application represented by this row. |
| SingleRunTimeHHMMSS | Gives the time to run the application in “Once” schedule mode |
| Enabled | (1 = Yes, 2 = No) – designates whether this row should be processed – can be used to cause the OmniSystemSchedule to ignore processing this row. |