Matisse® OEM Packaging Guide

January 2017

Matisse OEM Packaging Guide

PDF generated 7 January 2017

Contents

1   Introduction

1.1 Scope of this document

This document is intended to help Matisse Original Equipment Manufacturers (OEM) learn the requirements and techniques to include Matisse binaries in their own installation package.

This document covers the content of a Matisse standard delivery, the minimum binaries to be included in an OEM application. The pre- and post- installation operations as well as un-installation operations are also described in great details.

If there is anything you would like to see added, or if you have any questions about or corrections to this document, please send e-mail to support@matisse.com.

1.2 Before Reading this document

Throughout this document, we presume that you already know Matisse, as well as the installation builder tool of your choice (i.e. InstallShield for Windows and RPM for Linux).

2   OEM Delivery

Unlike the developers version which contains all the tools and utilities to design and develop applications on top of Matisse, the delivery for an OEM application consists in a handful of binaries and dynamic libraries referred as Matisse runtime.

CAUTION: The development tools and utilities, which constitute the development kit, cannot be delivered in an OEM package.

3   Installation Content

There is a minimum of command files and libraries that you need to deliver with your OEM package.

3.1 Runtime Client-Server Files

Server Files

There is a minimum of 5 binary files that are mandatory to run Matisse in an OEM application.

Filenames	Comments
bin\mts.exe	Mandatory – Matisse server.
bin\mt_backup.exe	Recommended – command line backup utility.
bin\mt_connection.exe	Recommended – command line connection management utility.
bin\mt_dts.exe	Optional – command line data transformation management utility.
bin\mt_emgr.exe	Optional – enterprise manager graphical tool
bin\mt_file.exe	Recommended – command line datafiles management utility.
bin\mt_msms.exe	Mandatory – Windows Service – Matisse Server Manager.
bin\mt_partition.exe	Optional – command line disk partition management utility.
bin\mt_portmon.exe	Mandatory – Windows Service – Matisse Port monitor.
bin\mt_pmadm.exe	Mandatory – command line port monitor administration utility.
bin\mt_replicate.exe	Optional – command line database replication management utility.
bin\mt_sdl.exe	Optional – command line schema management utility.
bin\mt_server.exe	Mandatory – command line server manager utility.
bin\mt_smlistener.exe	Optional – Windows Service – Matisse Remote Administration.
bin\mt_sql.exe	Optional – command line SQL query utility.
bin\mt_transaction.exe	Recommended – command line active transaction management utility.
bin\mt_user.exe	Optional – command line database user management utility.
bin\mt_version.exe	Optional – command line database snapshot management utility.
bin\mt_xml.exe	Optional – command line XML import/export utility
bin\mt_xsr.exe	Optional – command line XML-bsaed Snapshot Replication utility
lib\sqlhelp.txt	Optional – help for the command line SQL query utility.

Client Files

There is only one mandatory library necessary to run Matisse in an OEM application.

Filenames	Comments
bin\matisse.dll	Mandatory – Matisse client library.
bin\matisseCOM.dll	Optional – Matisse COM API library.
bin\matisseDTS.dll	Optional – Matisse Data Transformation API library.
bin\matisseEMGR.dll	Optional – Matisse Enterprise Manager API library.
bin\matisseJAVA.dll	Optional – Matisse Java Native library.
bin\matisseSDL.dll	Optional – Matisse Schema Management API library.
bin\matisseXML.dll	Optional – Matisse XML Import/Export API library
bin\MatisseNet.dll	Optional – Matisse .NET library.
lib\matisse.jar	Optional – Matisse Java Binding JAR.

NOTE: The Eiffel, PHP and Python language bindings are open-source and can be freely redistributed with the OEM delivery.

3.2 Runtime Embedded Files

There is only one mandatory library necessary to run Matisse Embedded in an OEM application.

Filenames	Comments
bin\matisselite.dll	Mandatory – Matisse embedded (serverless) library.
bin\matisseliteDTS.dll	Optional – Matisse Data Transformation API library.
bin\matisseliteEMGR.dll	Optional – Matisse Enterprise Manager API library.
bin\matisseliteJAVA.dll	Optional – Matisse Java Native library.
bin\matisseliteSDL.dll	Optional – Matisse Schema Management API library.
bin\matisseliteXML.dll	Optional – Matisse XML Import/Export API library
bin\MatisseLiteNet.dll	Optional – Matisse .NET library.
lib\matisselite.jar	Optional – Matisse Java Binding JAR.
bin\mt_dts_lite.exe	Optional – command line data transformation management utility.
bin\mt_emgr_lite.exe	Optional – enterprise manager graphical tool
bin\mt_file_lite.exe	Optional – command line datafiles management utility.
bin\mt_sdl_lite.exe	Optional – command line schema management utility.
bin\mt_server_lite.exe	Optional – command line server manager utility.
bin\mt_sql_lite.exe	Optional – command line SQL query utility.
bin\mt_user_lite.exe	Optional – command line database user management utility.
bin\mt_version_lite.exe	Optional – command line database snapshot management utility.
bin\mt_xml_lite.exe	Optional – command line XML import/export utility
lib\sqlhelp.txt	Optional – help for the command line SQL query utility.

NOTE: The Eiffel, PHP and Python language bindings are open-source and can be freely redistributed with the OEM delivery.

3.3 Configuration Files

You need to provide the Matisse configuration file(s) necessary to run the OEM application.

Filenames	Comments
config\<mydb>.cfg	Mandatory – the configuration file for your database server. Make sure the parameters values are adequate with your application requirements.
config\mtlk.dat	Mandatory – the OEM license key file for your Matisse Client Server application.
config\mtlk_lite.dat	Mandatory – the OEM license key file for your Matisse embedded application.

NOTE: The Matisse OEM license key is per OEM application and identify the OEM application.

3.4 Database Files

You may also provide the Matisse dafiles for your application preloaded with data to run the OEM application.

Filenames	Comments
datafiles\<mydb>.dat	Optional – One or multiple datafiles which contains the initial data for your application.

4 Including initial data

Depending on the time of Matisse utilities you want to deliver with your package, we suggest 2 options to provide initial data with your application.

4.1 Creating a preloaded datafile

You may want to include in your delivery a datafile that contains the initial data necessary to run your application. We recommend you create a datafile with the minimal size necessary to contain your data. Then when your application starts the database server, it can extend the datafile as necessary or simply set the `AUTOEXTEND` option in the database configuration file.

4.2 Creating a loading script

You can also create a script that loads your database schema and some initial data.

You can provide your database schema in ODL or SQL DDL format and the initial data in XML or CSV format. You need to provide in the delivery the appropriate Matisse utilities to load the schema and the data.

5   System Requirements

5.1 Hardware

- At least 64 MB of RAM
- At least 10M bytes of free disk space: 3M bytes for the binary files and a minimum of 7M bytes for the datafiles of your database

5.2 Software

- Support for Windows (2003/2008/2012/Vista/7/8) on systems based on x86 (32-bit) and x86_64 (64-bit) chips families.
- Support for Red Hat-based and Debian-based Linux distributions on x86 (32-bit) and x86_64 (64-bit) chips families.
- Support for Solaris 10/11 on SPARC with 32-bit kernel and 64-bit kernel.
- Support for Solaris 10/11 on x86 (32-bit and 64-bit) chips families.

6   Building the installation package

6.1 Pre-installation operations

By default, the installation of Matisse does not require any pre-installation operations.

However if you plan to provide the Matisse Enterprise Manager and/or the Matisse SMListener Service with your package, you need to check that Java JRE 1.7 is already installed on the machine and report an installation error if not present or install it.

6.2 Post-installation operations

Updating the registry

Matisse relies on the definition of a few keys to work properly.

MATISSE key

Under `HKEY_LOCAL_MACHINE\SOFTWARE`, create the `MATISSE` key.

Matisse values

Under the `MATISSE` key, create the four Matisse values of type `REG_SZ` and specify their value as described in the table below:

Name	Type	Data
MATISSE_HOME	REG_SZ	<installdir>
MATISSE_CFG	REG_SZ	<installdir>\config
MATISSE_LOG	REG_SZ	<installdir>\log
MATISSE_DDL_PATH	REG_SZ	<installdir>\bin;<installdir>\Java\jre6\bin;<installdir>\Java\jre6\bin\client
MATISSE_PORTMON_ADDR	REG_SZ	tcp-7421
MATISSE_SMLISTENER_ADDR	REG_SZ	7412

Under the `HKEY_CURRENT_USER\Environment` key, create the `MATISSE_HOME` value of type `REG_SZ` and specify its value to `<installdir>`

Updating the path

Update the `PATH` to make Matisse executables and DLLs accessible from your application.

```
PATH=%PATH%;<installdir>\bin
```

NOTE: If you plan to include Matisse Enterprise Manager and/or the Matisse Remote Administration Listener Service in your delivery, you need to make sure Java JRE 1.7 is installed on the machine and add the path of the java command and the JVM library to the path.

```
PATH=%PATH%;<installdir>\bin;;<installdir>\Java\jre7\bin;<installdir>\Java\jre7\bin\client
```

Installing Matisse Services

Matisse Port Monitor

Run the following command to install the service:

```
bin\mt_portmon -install
```

If the `MATISSE_Portmon` service already exists, you need to remove it, running the following command:

```
bin\mt_portmon –remove
```

Matisse Server Manager

Run the following command to install the service:

```
bin\mt_msms -install
```

If the `MATISSE_MSMS` service already exists, you need to remove it, running the following command:

```
bin\mt_msms -remove
```

Matisse SMListener

Run the following command to install the service:

```
bin\mt_smlistener -install
```

If the `MATISSE_SMLISTENER` service already exists, you need to remove it, running the following command:

```
bin\mt_smlistener -remove
```

6.3 Uninstall operations

Removing Matisse Services

Matisse Port Monitor

Run the following command to remove the service:

```
bin\mt_portmon -remove
```

Matisse Server Manager

Run the following command to remove the service:

```
bin\mt_msms -remove
```

Matisse SMListener
Run the following command to remove the service:

```
bin\mt_smlistener -remove
```

7 Testing the installation package

Select a machine where Matisse has never been installed before and run your installation package.

Step 1

Run the module of your application that dialogs with the Matisse database.

Step 2

Then uninstall the package and make sure that the Matisse binaries in bin and lib are properly removed. Then check that the Matisse Services have been properly removed as well.

8 Troubleshooting the installation package

8.1 Checking the PATH

Make sure that the `PATH` has been correctly updated.

A simple test consists in starting a Windows command line and executing a Matisse command (i.e. `mt_server –V`).
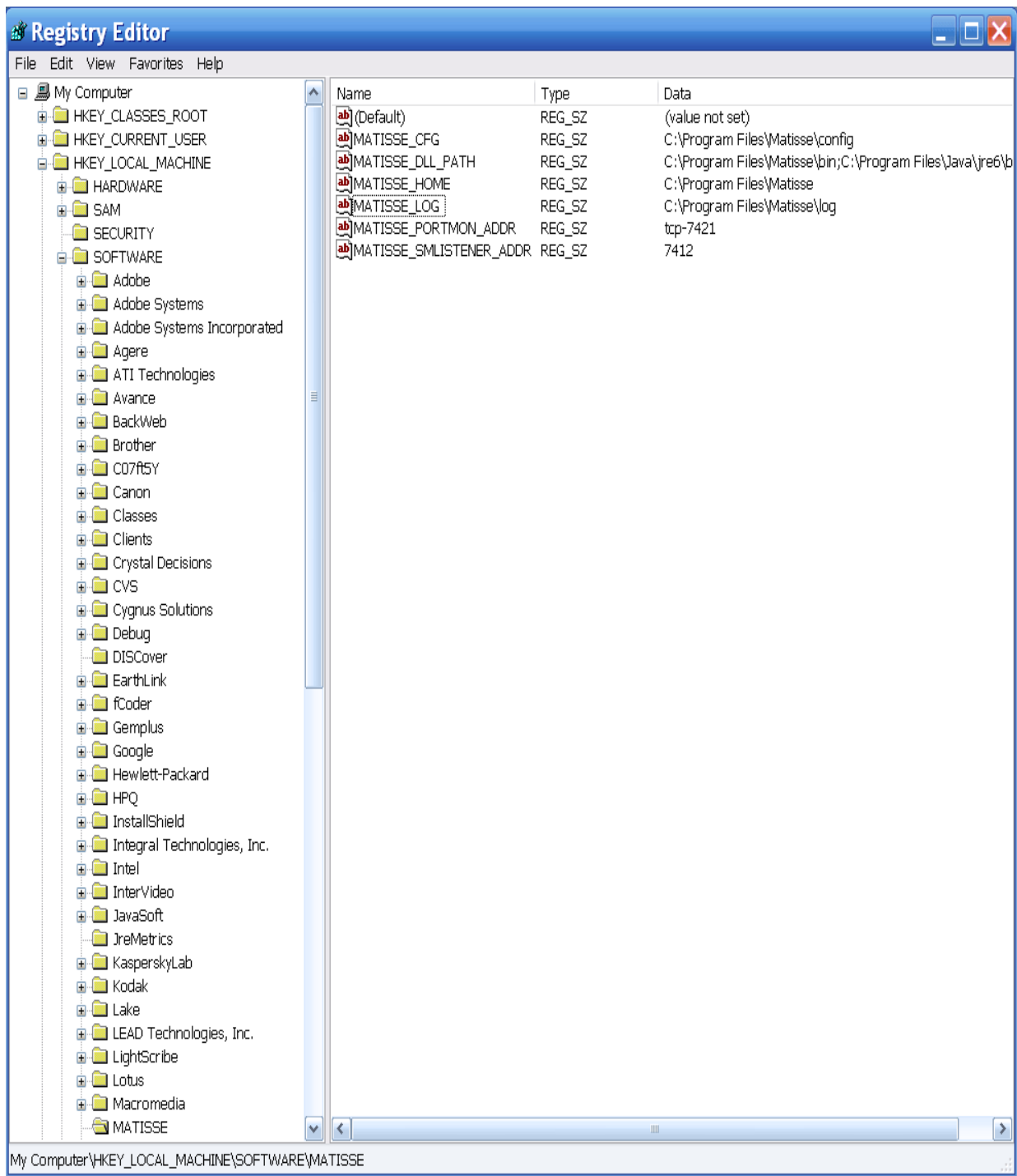
```
C:\>mt_server -V
MATISSE Server Manager x64 Version 9.1.0.0 (64-bit Edition) - Apr 10
    2013.
(c) Copyright 2013 Matisse Software Inc. All rights reserved.
TOPAZ - Windows XP 5.1 SP3 32-bit - i586 - 1 CPU - 1982Mbytes -
    001122334455
```

If the command does not execute properly, list the environment variable with the command set.

```
C:\>set
[...]
Path=C:\Program Files\Microsoft Visual Studio 8\Common7\IDE;[...]
;C:\Program Files\Matisse\bin;C:\Program Files\Java\jre7\bin;
C:\Program Files\Java\jre7\bin\client;[...]
```
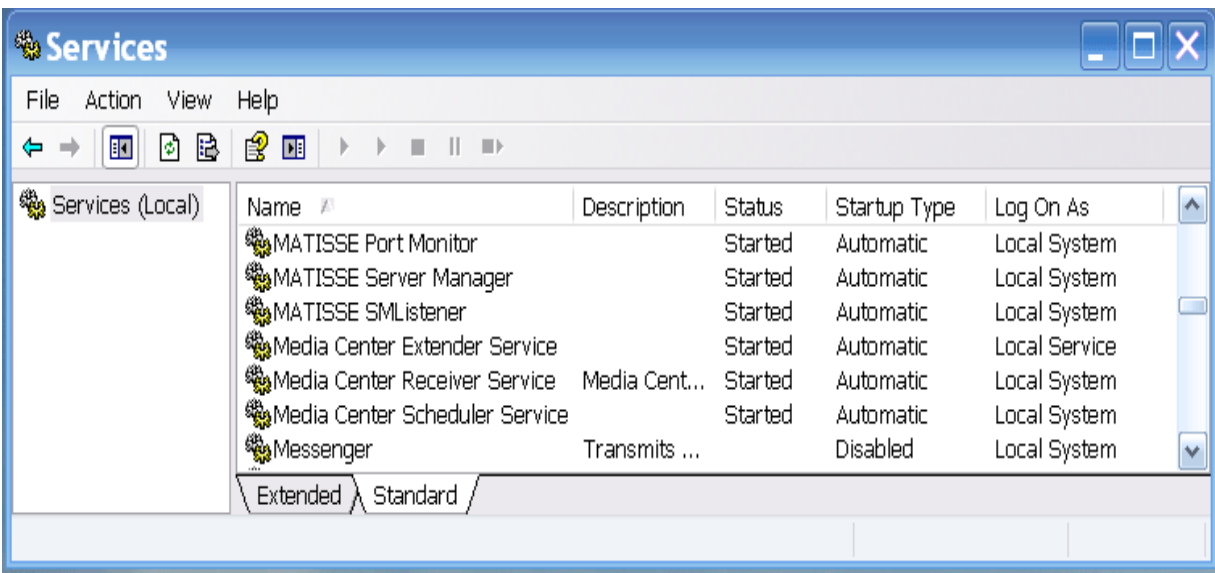
8.2 Checking the Registry

Check that the registry contains a `MATISSE` key under `HKEY_LOCAL_MACHINE\SOFTWARE` and the `MATISSE_HOME`, `MATISSE_CFG`, `MATISSE_LOG` and `MATISSE_DLL_PATH` exists and are set to the expected values.

8.3 Checking the Services

Check that the Matisse services are up and running.

8.4 Checking the Port Monitor

Check that the port monitor is fully accessible from your environment and that your database is properly registered.

```
C:\>mt_pmadm -L
31 May. 2012 14:56:02 PMADM-I-PMSTATE, Port monitor is enabled
31 May. 2012 14:56:02 PMADM-I-TRPTYPE, Transport type: tcp
31 May. 2012 14:56:02 PMADM-I-SVCLIST, Registered services:
EXAMPLE
```

8.5 Checking the Database Server

Check that the database server can be started and stopped properly.

Starts a Windows command line and executes the following commands:

```
C:\>mt_server list
1 transport(s), 1 server(s)
             mttcp
EXAMPLE       X

C:\>mt_server -d example stop

C:\>mt_server list
1 transport(s), 0 server(s)
             mttcp
```