JobScheduler

Installation by Copying

Deployment of multiple JobSchedulers on distributed servers by copying a template JobScheduler

March 2015

Contact Information

Software- und Organisations-Service GmbH

Giesebrechtstr. 15
D-10629 Berlin
Germany

Telephone +49 (0)30 86 47 90-0
Telefax +49 (0)30 8 61 33 35
Mail info@sos-berlin.com
Web http://www.sos-berlin.com

Last Updated: 03/13/2015 12:03 PM

This documentation is based on JobScheduler Version 1.7.4169.

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

We would appreciate any feedback you have, or suggestions for changes and improvements; please forward your comments to info@sos-berlin.com.

Table of Contents

1 Introduction

This document describes how to install a JobScheduler by copying/cloning a template JobScheduler. This procedure should assist the deployment of JobSchedulers on multiple servers.

You start this procedure by installing the JobScheduler that will be used as the template for the others using the normal setup procedure. This JobScheduler should include the Java® Runtime Environment (JRE). Note that the copied/cloned JobSchedulers will use the same database as the template. It is important to ensure that the clones have a unique JobScheduler-ID and use the template Java® Runtime Environment (JRE). In addition, some modifications have to be made to the template JobScheduler's file system: the modifications required depend on the operating system:

On Microsoft® Windows® systems a script has to be started after copying the template in order to install the service and to generate the shortcuts and symlinks.

On Linux® systems the cloned JobScheduler should have the same user as the template JobScheduler, nothing else needs to be configured after cloning.

2 Install Java

A version of Oracle® Database's Java® Runtime Environment (JRE) 7 that can be installed on both Microsoft® Windows® and Linux® systems without using a setup (.exe) or rpm call is used. Such Java® Runtime Environment (JRE) versions are available as archives and only need to be unpacked into a suitable folder. They can be downloaded from Oracle® Database's web-site and can be easily recognised by their ".tar.gz" extension (see Oracle - JRE Downloads). Such a tar.gz Java® Runtime Environment (JRE) should be "installed" alongside the template JobScheduler so that the Java® Runtime Environment (JRE) can be included in the template JobScheduler during cloning.

At this stage in the cloning procedure, simply unpack the tar.gz Java® Runtime Environment (JRE) archive to any folder on the template JobScheduler server. Ensure that Java® Runtime Environment (JRE) version you download has the same number of bits (i.e. 32 or 64) as the template JobScheduler.

3 Installation of the "template" JobScheduler

Install the template JobScheduler using the standard JobScheduler setup. See the Installation Guide.

3.1 Installation path

Two directories are set during the JobScheduler installation procedure (`$SCHEDULER_HOME` and `$SCHEDULER_DATA`). These have to be KEPT IDENTICAL for the installation of the template JobScheduler. Note that `$SCHEDULER_HOME` will be used in the rest of this document to refer to both installation directories.

3.1.1 Linux®

We recommend that you create a user such as "jobscheduler" to carry out the installation. Use this user - without root permissions - (`./setup.sh -u`) and set the `$HOME` directory as the installation directory.

If the tar.gz Java® Runtime Environment (JRE) described above is the only Java® Runtime Environment (JRE) on the template JobScheduler server, then you will need to set the `PATH` environment variable before starting the set-up at the command line.

```
> export PATH=/path/to/java/bin:$PATH
> ./setup.sh -u
```

Example: Set $PATH before setup on Linux®

Note that the Java® Runtime Environment (JRE) used during the setup will not affect the use of the tar.gz Java® Runtime Environment (JRE) by the template and clone JobSchedulers later on.

3.1.2 Microsoft® Windows®

Do not install the JobScheduler in the `Program Files` folder, as files will have to be modified after installation. Use of the `Program Files` folder makes modifications more difficult as administrator permissions are required to make changes to files installed in this folder. In addition, after installation the JobScheduler may have insufficient permissions for writing, for example, log files.

If the tar.gz Java® Runtime Environment (JRE) described above is the only Java® Runtime Environment (JRE) on the template JobScheduler server, then you will need to set the `Path` environment variable before starting the set-up at the command line.

```
> set Path=c:\path\to\java\bin;%Path%
> setup.cmd
```

Example: Set %Path% before setup on Microsoft® Windows®

Note that the Java® Runtime Environment (JRE) used during the setup will not affect the use of the tar.gz Java® Runtime Environment (JRE) by the template and clone JobSchedulers later on.

3.2 Database

All the cloned JobSchedulers will use the same database. Configure the database on the template JobScheduler server before installing the template JobScheduler and enter the necessary database connection as described in the database configuration chapter of our JobScheduler Installation Guide.

3.3 JobScheduler-Port, Jetty-Ports

All cloned JobSchedulers will use the same ports later on. The default port for JobScheduler is 4444 and the default for Jetty is 44440 for http and 8443 for https. It is assumed that no other JobScheduler or other applications that use these ports will be installed on the target servers. Note that if these default ports are not free, it is important to ensure that the ports selected are available on all target servers.

4 Prepare the Template JobScheduler for Cloning

A number of simple manual changes are required before the template JobScheduler can be cloned. The clones require a unique JobScheduler ID, a flexible installation path and the "correct" Java® Runtime Environment (JRE). This means that the Java® Runtime Environment (JRE) described above has to be added to the installation directory and either 1 or 2 (depending on the OS) files are modified. The files to be modified are:

- `$SCHEDULER_HOME/config/sos.ini` (for Microsoft® Windows®)
- `$SCHEDULER_HOME/user_bin/jobscheduler_environment_variables.cmd` (for Microsoft® Windows®)
- `$SCHEDULER_HOME/user_bin/jobscheduler_environment_variables.sh` (for Linux®)

4.1 Add the Java® Runtime Environment (JRE)

In order to include the Java® Runtime Environment (JRE) with the template JobScheduler, the Java® Runtime Environment (JRE) should either be moved or unpacked in the `$SCHEDULER_HOME` folder. You will now have a `$SCHEDULER_HOME/jre[version]` directory (for example: `$SCHEDULER_HOME/jre1.7.0_40`). Rename this directory `$SCHEDULER_HOME/jre`, to enable you to update the Java® Runtime Environment (JRE) environment without having to change anything in the other files of the template JobScheduler.

4.2 Set up the ./config/sos.ini for Microsoft® Windows®

JobScheduler normally uses the registry to find the JVM. In addition, the `$SCHEDULER_HOME/config/sos.ini` file should be set so that JobScheduler uses the JVM in the `$SCHEDULER_HOME/jre`. This setting is to be found in the [java] section of the `$SCHEDULER_HOME /config/sos.ini` file

```
vm = $SCHEDULER_HOME/jre/bin/client/jvm.dll
```

Example: $SCHEDULER_HOME/config/sos.ini

4.3 Set up the ./user_bin/jobscheduler_environment_variables.(sh|cmd)

Copy the `$SCHEDULER_HOME/bin/jobscheduler_environment_variables.(sh|cmd)` to the `$SCHEDULER_HOME /user_bin/`. In this file you can configure a unique ID, a flexible installation path and the `JAVA_HOME` environment variable for JobScheduler, JobScheduler Object Editor (JOE) and JobScheduler Information Dashboard (JID). The flexible installation path ensures that the clone can be copied to any required location.

4.3.1 Linux®

Modify the $SCHEDULER_HOME/user_bin/jobscheduler_environment_variables.sh as follows:

```
  ...
  USER=`whoami`
 SCHEDULER_USER=jobscheduler

  # Installationpath is the parent folder of this script
 SCHEDULER_HOME="`dirname $0`"
 SCHEDULER_HOME="`dirname $SCHEDULER_HOME`"

 # SCHEDULER_HOME and SCHEDULER_DATA are identically
 SCHEDULER_DATA="$SCHEDULER_HOME"

 # JobScheduler ID has to be unique
 SCHEDULER_ID="$HOSTNAME.4444"

 # Set JAVA_HOME
 JAVA_HOME="$SCHEDULER_HOME/jre"

 # The following can retain unchanged

 SH="/bin/sh -c"

LD_LIBRARY_PATH="$SCHEDULER_HOME/lib:/usr/local/lib:$JAVA_HOME/lib/i386:$JAVA_HOME/lib/i386/client:$JAVA_HOME/jre/lib/i
386:$JAVA_HOME/jre/lib/i386/client:$LD_LIBRARY_PATH"

 SOS_INI="$SCHEDULER_DATA/config/sos.ini"
 SCHEDULER_PID="$SCHEDULER_DATA/logs/scheduler.pid"
 SCHEDULER_INI="$SCHEDULER_DATA/config/factory.ini"
 SCHEDULER_CLUSTER_OPTIONS=""
 SCHEDULER_PARAMS="-service \"-id=$SCHEDULER_ID\" \"-sos.ini=$SOS_INI\" \"-ini=$SCHEDULER_INI\"
\"-config=$SCHEDULER_DATA/config/scheduler.xml\" \"-param=$SCHEDULER_DATA\" \"-cd=$SCHEDULER_DATA\"
\"-include-path=$SCHEDULER_DATA\""
 SCHEDULER_START_PARAMS="$SCHEDULER_PARAMS \"-log-dir=$SCHEDULER_DATA/logs\" \"-pid-file=$SCHEDULER_PID\""
 SCHEDULER_BIN="$SCHEDULER_HOME/bin/scheduler"
 SCHEDULER_SAFE="$SCHEDULER_HOME/bin/scheduler_safe.sh"

 export SCHEDULER_USER
 export SCHEDULER_HOME
 export SCHEDULER_DATA
 export SOS_INI
 export LD_LIBRARY_PATH
 export JAVA_HOME
```

Example: $SCHEDULER_HOME/user_bin/jobscheduler_environment_variables.sh

The unique dynamic ID is set using SCHEDULER_ID="$HOSTNAME.4444", where 4444 is the port of the template JobScheduler. This port can be found in the template JobScheduler's $SCHEDULER_HOME/config/scheduler.xml file.

The $SCHEDULER_HOME/user_bin/jobscheduler_environment_variables.sh has to be executable. If necessary then type

```
> chmod a+x ./user_bin/jobscheduler_environment_variables.sh
```

Example: set $SCHEDULER_HOME/user_bin/jobscheduler_environment_variables.sh executable

4.3.2 Microsoft® Windows®

Change the $SCHEDULER_HOME/user_bin/jobscheduler_environment_variables.cmd, as follows:

```
...
:begin

rem Installationpath is the parent folder of this script
set INSTALL_PATH=%~dp0
set INSTALL_PATH=%INSTALL_PATH:\user_bin\=%

rem SCHEDULER_HOME and SCHEDULER_DATA are identically
set APPDATA_PATH=%INSTALL_PATH%

set SCHEDULER_HOME=%INSTALL_PATH:\=/%
set SCHEDULER_DATA=%APPDATA_PATH:\=/%

rem JobScheduler ID has to be unique
set SCHEDULER_ID=%COMPUTERNAME%.4444

rem Set JAVA_HOME
set JAVA_HOME=%INSTALL_PATH%\jre

rem The following can retain unchanged

set SOS_INI=%SCHEDULER_DATA%/config/sos.ini
set SCHEDULER_INI=%SCHEDULER_DATA%/config/factory.ini
set SCHEDULER_PID="%SCHEDULER_DATA%/logs/scheduler.pid"
set SCHEDULER_CLUSTER_OPTIONS=
set SCHEDULER_PARAMS=-id="%SCHEDULER_ID%" -sos.ini="%SOS_INI%" -config="%SCHEDULER_DATA%/config/scheduler.xml"
-ini="%SCHEDULER_INI%" -env="SCHEDULER_HOME=%SCHEDULER_HOME%" -env="SCHEDULER_DATA=%SCHEDULER_DATA%"
-param="%SCHEDULER_DATA%" -cd="%SCHEDULER_DATA%" -include-path="%SCHEDULER_DATA%"
set SCHEDULER_START_PARAMS=%SCHEDULER_PARAMS% -log-dir="%SCHEDULER_DATA%/logs" -pid-file=%SCHEDULER_PID%
set SCHEDULER_BIN="%INSTALL_PATH%\bin\scheduler.exe"
```

Example: $SCHEDULER_HOME/user_bin/jobscheduler_environment_variables.cmd

"set SCHEDULER_ID=%COMPUTERNAME%.4444" is used to set a unique, dynamic ID, where 4444 is the port of the template JobScheduler. This port can be determined from the $SCHEDULER_HOME/config/scheduler.xml file.

5 Cloning

After the changes described above have been made to the template JobScheduler , the whole $SCHEDULER_HOME directory can be packed in an archive and copied to another computer.

The $SCHEDULER_HOME/db and $SCHEDULER_HOME/Uninstaller can be optionally deleted from the template JobScheduler. The uninstaller will not work in cloned JobSchedulers: cloned JobSchedulers have to be deleted manually (see JobScheduler Installation Guide). The $SCHEDULER_HOME/logs folder can also be emptied beforehand.

Note that the clone requires read, write and execute permissions in the installation directory.

5.1 Linux®

On Linux® systems the archive has to be unpacked by the user (e.g. "jobscheduler") that will use the JobScheduler. Note that a user with this name has to be configured on the target computer before the cloned JobScheduler is installed - as already described for the template JobScheduler. In addition, the clone has to be copied in the correct $HOME directory.

The SCHEDULER_USER can be modified after cloning if required by modifying the $SCHEDULER_HOME/user_bin/jobscheduler_environment_variables.sh file as follows:

```
SCHEDULER_USER=[otherUser]
```

Example: $SCHEDULER_HOME/user_bin/jobscheduler_environment_variables.sh

5.2 Microsoft® Windows®

The $SCHEDULER_HOME/install/create_service_symlinks_shortcuts.cmd script file should be executed after unpacking the archive. This script creates the necessary symlinks, checks the Java® Runtime Environment (JRE) environment, creates shortcuts and installs and starts the JobScheduler service.

Now you're ready!