Working with Software AG Runtime

Version 9.8

April 2015

This document applies to webMethods Product Suite Version 9.8 and to all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Copyright © 1999-2015 Software AG, Darmstadt, Germany and/or Software AG USA Inc., Reston, VA, USA, and/or its subsidiaries and/or its affiliates and/or their licensors.

The name Software AG and all Software AG product names are either trademarks or registered trademarks of Software AG and/or Software AG USA Inc. and/or its subsidiaries and/or its affiliates and/or their licensors. Other company and product names mentioned herein may be trademarks of their respective owners.

Detailed information on trademarks and patents owned by Software AG and/or its subsidiaries is located at http://softwareag.com/licenses.

This software may include portions of third-party products. For third-party copyright notices, license terms, additional rights or restrictions, please refer to "License Texts, Copyright Notices and Disclaimers of Third Party Products". For certain specific third-party license restrictions, please refer to section E of the Legal Notices available under "License Terms and Conditions for Use of Software AG Products / Copyright and Trademark Notices of Software AG Products". These documents are part of the product documentation, located at http://softwareag.com/licenses and/or in the root installation directory of the licensed product(s).

Use, reproduction, transfer, publication or disclosure is prohibited except as specifically provided for in your License Agreement with Software AG.

Document ID: WEBM-RUNTIME-98-20150415

Table of Contents

About this Guide **5**
Document Conventions 5
Online Information 6

Concepts **7**
Overview 8
Architecture 8

Configuring Runtime **11**
Working with HTTP Connectors 12
Using the Predefined HTTP Connector 12
Configuring a User-Defined HTTP Connector 12
Working with HTTPS Connectors 13
Using the Predefined HTTPS Connector 13
Configuring a User-Defined HTTPS Connector 14
Server-Side Configuration 15
Securing the HTTPS Connector Password 15
Client-Side Configuration 16
Working with JMX Connectors 16
Using the Predefined JMX Connector 16
Configuring the Java Service Wrapper 17
The Java Service Wrapper Configuration Files 18
Configuring Wrapper Java Properties 18
Configuring Memory Properties 19
Configuring Wrapper Logging Properties 20
Fault Monitoring 24
Generating a Thread Dump 24
Configuring the Runtime Log Settings 24
Hot Configuration Update 25
Using Path Tokens 25

Starting and Stopping Runtime **27**
Starting and Stopping Runtime 28
Starting and Stopping Runtime in Windows 28
Starting and Stopping Runtime in UNIX 28

Securing Runtime **31**
Managing the Runtime Security 32

Completing the Software AG Runtime Installation **33**
Post-Installation Steps 34

About this Guide

This guide explains how to configure, manage, and use Software AG Runtime.

Document Conventions

Convention	Description
Bold	Identifies elements on a screen.
Narrowfont	Identifies storage locations for services on webMethods Integration Server, using the convention *folder.subfolder:service* .
UPPERCASE	Identifies keyboard keys. Keys you must press simultaneously are joined with a plus sign (+).
Italic	Identifies variables for which you must supply values specific to your own situation or environment. Identifies new terms the first time they occur in the text.
`Monospace font`	Identifies text you must type or messages displayed by the system.
{ }	Indicates a set of choices from which you must choose one. Type only the information inside the curly braces. Do not type the { } symbols.
|	Separates two mutually exclusive choices in a syntax line. Type one of these choices. Do not type the | symbol.
[ ]	Indicates one or more options. Type only the information inside the square brackets. Do not type the [ ] symbols.
...	Indicates that you can type multiple options of the same type. Type only the information. Do not type the ellipsis (...).

Online Information

Software AG Documentation Website

You can find documentation on the Software AG Documentation website at http://documentation.softwareag.com. The site requires Empower credentials. If you do not have Empower credentials, you must use the TECHcommunity website.

Software AG Empower Product Support Website

You can find product information on the Software AG Empower Product Support website at https://empower.softwareag.com.

To submit feature/enhancement requests, get information about product availability, and download products, go to Products.

To get information about fixes and to read early warnings, technical papers, and knowledge base articles, go to the Knowledge Center.

Software AG TECHcommunity

You can find documentation and other technical information on the Software AG TECHcommunity website at http://techcommunity.softwareag.com. You can:

- Access product documentation, if you have TECHcommunity credentials. If you do not, you will need to register and specify "Documentation" as an area of interest.
- Access articles, code samples, demos, and tutorials.
- Use the online discussion forums, moderated by Software AG professionals, to ask questions, discuss best practices, and learn how other customers are using Software AG technology.
- Link to external websites that discuss open standards and web technology.

1 Concepts

- Overview 8
- Architecture 8

Overview

Software AG Runtime is an runnable instance of the Software AG Common Platform environment. The Common Platform is OSGi-based and offers the possibility to dynamically construct executable instances of various products.

It is assembled dynamically to host and run web applications. Software AG Runtime is available in your Software AG installation under *Software AG_directory*/profiles/CTP.

Software AG Runtime comprises of the following key components:

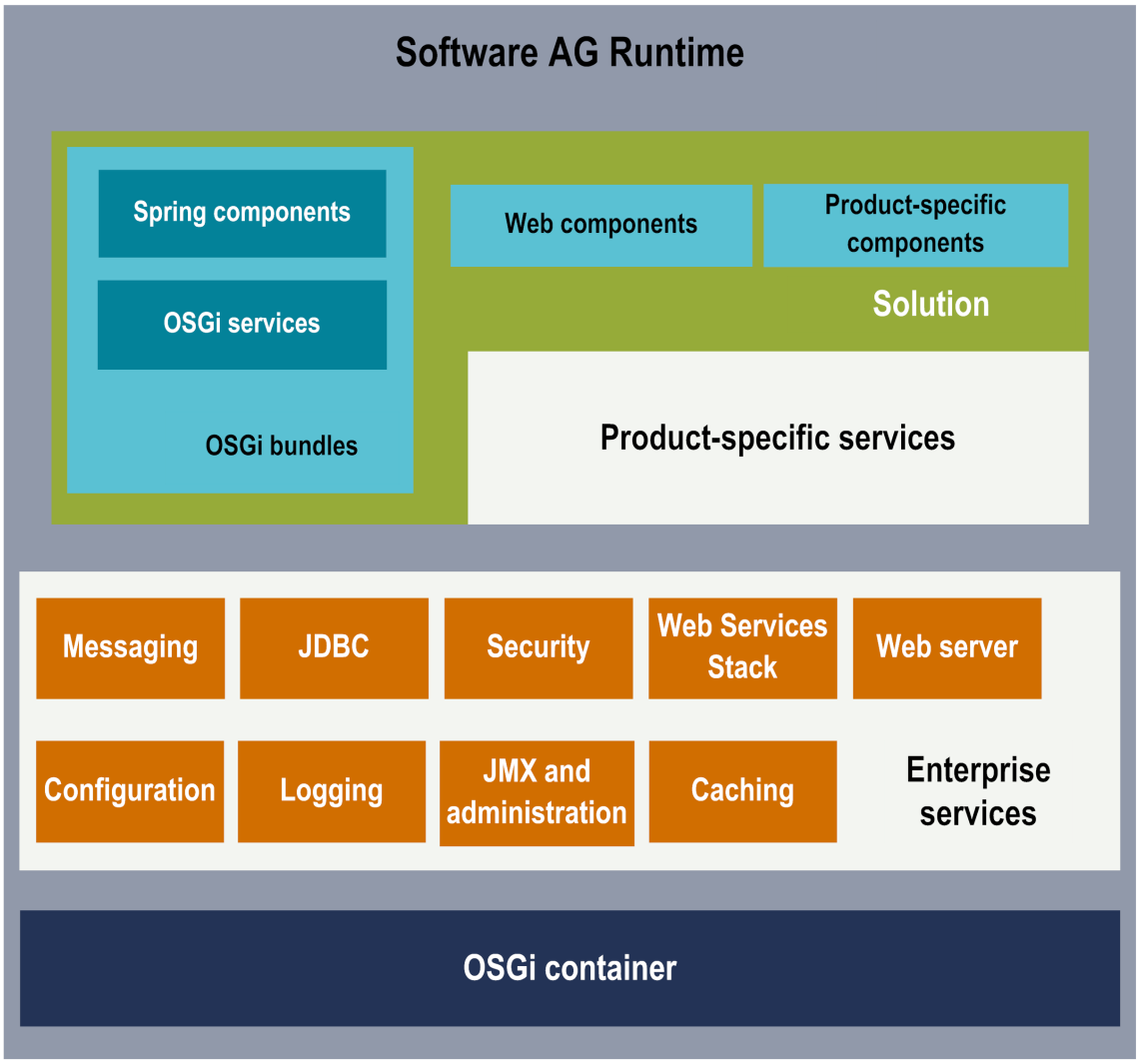
- Software AG Web Server based on Apache Tomcat
- Software AG Security Infrastructure
- Software AG NERV
- Software AG Web Services Stack
- Spring Framework

Software AG System Management Hub, webMethods EntireX, and CentraSite use the Software AG Runtime instance of the Software AG Common Platform environment. Terracotta BigMemory client libraries are integrated with Software AG Runtime and are therefore available to these products. Other Software AG products have their own instances in other profile directories.

For more information about your product's runtime instance, see the product documentation.

Architecture

You can use the OSGi technology which provides modular architecture to construct your applications out of different reusable components, and then run them within the Software AG Runtime instance. As an instance of the OSGi-based Common Platform, Software AG Runtime is assembled dynamically and represents a container for your web applications.


Software AG Runtime
Spring components
OSGi services
OSGi bundles
Web components
Product-specific components
Solution
Product-specific services
Messaging
JDBC
Security
Web Services Stack
Web server
Configuration
Logging
JMX and administration
Caching
Enterprise services
OSGi container

2 Configuring Runtime

- Working with HTTP Connectors 12
- Working with HTTPS Connectors 13
- Working with JMX Connectors 16
- Configuring the Java Service Wrapper 17
- Configuring the Runtime Log Settings 24
- Hot Configuration Update 25
- Using Path Tokens 25

Working with HTTP Connectors

Using the Predefined HTTP Connector

Software AG Runtime is installed with a predefined HTTP connector. It is defined in the com.softwareag.catalina.connector.http.pid-*port_number* .properties file, located in the *Software AG_directory*\profiles\CTP\configuration\com.softwareag.platform.config.propsloader directory. You can use the predefined HTTP connector as is, or you can modify some of its properties.

Configuring a User-Defined HTTP Connector

You can configure one or more user-defined connectors to use instead of the predefined connector.

You can configure a user-defined HTTP connector by:

- Modifying the predefined HTTP connector settings in the com.softwareag.catalina.connector.http.pid-*port_number* .properties file. The file is available in the *Software AG_directory*\profiles\CTP\configuration\com.softwareag.platform.config.propsloader directory.
- Creating another properties file in the same location. The new configuration file must follow the naming convention of the predefined file by keeping the `com.softwareag.catalina.connector.https.pid-` prefix. The created file must have configured unique port and alias properties. The aliases of `defaultHttp` and `defaultHttps` are reserved for the predefined HTTP and HTTPS connectors and must not be used.
- Using the Software AG Command Central user interface. For more information, see the *Software AG Command Central Help*.

Note: Make sure that one of your HTTP connectors has the *alias* property set to `defaultHttp`. This value exists by default in the predefined HTTP connector definition. The `defaultHttp` value is used by the Common Platform to distinguish default connectors from other existing connectors. If this value is not present in at least one HTTP connector definition, this results in an invalid or corrupted Software AG Runtime configuration the next time you install a product in the profile.

To configure a user-defined HTTP connector

1. Browse to the *Software AG_directory* \profiles\CTP\configuration\com.softwareag.platform.config.propsloader directory.
2. In a text editor, open one of the following files:

- The existing com.softwareag.catalina.connector.http.pid-*port_number* .properties file to modify the predefined HTTP connector.

  Note: When using the existing HTTP connector, make sure you rename the properties file by modifying the *port_number* in the filename.

- The user-defined HTTP connector properties file to define a new HTTP connector.

3. Modify the values of the following properties:

Property	Description
`port`	Used to set the TCP port number on which the connector will create a server socket and await incoming connections.
`alias`	Used by Software AG Platform Manager to identify each connector. The default HTTP connector must have this property set to `defaultHttp` and this value must be unique and used for only one connector.
`enabled`	Used to specify whether the connector configuration is enabled. The default value is `false` and must be set to `true` to enable the connector.

4. Optionally, modify the other properties in the file. For more information about the available connector properties, see the Apache Tomcat documentation.
5. Save your modifications.

Working with HTTPS Connectors

Using the Predefined HTTPS Connector

Software AG Runtime is installed with a predefined HTTPS connector. It is defined in the com.softwareag.catalina.connector.https.pid-*port_number* .properties file, located in the *Software AG_directory*\profiles\CTP\configuration\com.softwareag.platform.config.propsloader directory.

To use the predefined HTTPS connector

1. Browse to the *Software AG_directory* \profiles\CTP\configuration\com.softwareag.platform.config.propsloader directory.

   Note: When using the existing HTTP connector, make sure you rename the properties file by modifying the *port_number* in the filename.

2. In a text editor, open the com.softwareag.catalina.connector.https.pid-*port_number* .properties file.
3. Modify the value of the `keystoreFile` property to point to a valid keystore file.

   The `keystoreFile` property points to the localhost_dont_use_in_production.jks keystore, located in the *Software AG_directory*\profiles\CTP\configuration\tomcat\conf. It is only a sample and must not be used for production purposes.
4. Modify the values of the following properties:

Property	Description
`port`	Used to set the TCP port number on which the connector will create a server socket and await incoming connections.
`alias`	Used by Software AG Platform Manager to identify each connector. The default HTTPS connector must have this property set to `defaultHttps` and this value must be unique and used for only one connector.
`enabled`	Used to specify whether the connector configuration is enabled. The default value is `false` and must be set to `true` to enable the connector.

5. Optionally, edit the other properties in the file. For more information about the available connector properties, see the Apache Tomcat documentation.
6. Save your modifications.

Note: Make sure that the *alias* property is set to `defaultHttps`. This is required so that Software AG Platform Manager is able to recognize this connector as the default HTTPS connector.

Configuring a User-Defined HTTPS Connector

You can configure one or more user-defined connectors to use instead of the predefined connector.

You can configure a user-defined HTTPS connector by:

- Modifying the predefined HTTPS connector settings in the com.softwareag.catalina.connector.https.pid-*port_number* .properties file. The file is available in the *Software AG_directory*\profiles\CTP\configuration\com.softwareag.platform.config.propsloader directory.
- Creating another properties file in the same location. The new configuration file must follow the naming convention of the predefined file by keeping the `com.softwareag.catalina.connector.https.pid-` prefix. The created file must

have configured unique port and alias properties. The aliases of `defaultHttp` and `defaultHttps` are reserved for the predefined HTTP and HTTPS connectors and must not be used.

- Using the Software AG Command Central user interface. For more information, see the *Software AG Command Central Help*.

Important: Make sure that one of your HTTPS connectors has the *alias* property set to `defaultHttps`. This value exists by default in the predefined HTTPS connector definition. The `defaultHttps` value is used by the Common Platform to distinguish default connectors from other existing connectors. If this value is not present in at least one HTTPS connector definition, this results in an invalid or corrupted Software AG Runtime configuration the next time you install a product in the profile.

Server-Side Configuration

Before you start the procedure for defining an HTTPS connector on the server side, make sure that you have your own server certificate. You must set the Common Name (CN) of the certificate to be identical to the URL of the server, without the "https://". For example, for a server at `https://MyWebServer:8443/`, the CN is `MyWebServer`.

To define an HTTPS connector on the server side

1. Browse to *Software AG_directory* \profiles\CTP\configuration\com.softwareag.platform.config.propsloader.
2. In a text editor, open one of the following files:
   - The existing com.softwareag.catalina.connector.https.pid-*port_number* .properties file to modify the predefined HTTPS connector.

     Note: When using the existing HTTPS connector, make sure you rename the properties file by modifying the *port_number* in the filename.

   - The user-defined HTTPS connector properties file to define a new HTTPS connector.
3. Define the location of the keystore file containing the server certificate by configuring its path to the `keystoreFile` property.
4. Set your password by specifying a value for the `keystorePass` property.
5. Set the Java keystore type using the `keystoreType` property. Software AG Runtime supports three types of Java keystores: `JKS` (default), `PKCS11` and `PKCS12`.
6. Save your modifications.

Securing the HTTPS Connector Password

The `keystorePass` property can be secured by adding a `@secure.` prefix to the property key. For example:

```
@secure.keystorePass=change_this_password
```

The next time the properties file configuration is loaded the value of the `keystorePass` property will be moved to an encrypted secure storage on the file system under the*Software AG_directory*\profiles\CTP\configuration\security\passman directory and the configuration will be written back replacing the value with a secure token that contains a handle from the secure storage instead of the original plaintext value.

You can change the value of the `keystorePass` property to change the password. If the password is already secured, to change the value of the `keystorePass` property you have to replace the secure token handle with a new plaintext password that will be secured in turn and will overwrite the previous password in the secure storage.

Client-Side Configuration

To accept an HTTPS connection on the client side

- Import the server certificate into your Internet browser truststore. In case of a PKI, import the CA certificate that issued the server certificate.

  Note: If you are accessing resources through a Web server's HTTPS protocol from a Java client using Oracle JSSE, you must also set a truststore via the *-Djavax.net.ssl.trustStore* property and a truststore password via the *-Djavax.net.ssl.trustStorePassword* property, for example:

  ```
  -Djavax.net.ssl.trustStore=<your_truststore_here>
  -Djavax.net.ssl.trustStorePassword=<your_truststore_password_here>
  ```

  Or

- When opening an HTTPS connection in your Internet browser, click **Yes** at the prompt if you trust the certificate.

Working with JMX Connectors

Using the Predefined JMX Connector

Software AG Runtime is installed with a predefined JMX connector. It is defined in the com.softwareag.jmx.connector.pid-port_number.properties file, located in the *Software AG_directory*\profiles\CTP\configuration \com.softwareag.platform.config.propsloader directory. The predefined JMX connector is used by Software AG Platform Manager to manage Software AG Runtime and it is not recommended to edit its settings unless specifically asked to do so. The following properties can be configured:

Property	Description
host	Specifies the name of local interface to listen on. If not specified the connector is open on all local interfaces.
port	Specifies the JMX port to listen on.
jaasRealm	Specifies the name of the JAAS entry in the JAAS configuration file to use for JMX authentication.
enabled	Specifies whether the connector is enabled.
permission	Specifies the name of the permission, which the authenticated user must have to allow a JMX connection to be set up. Using `*` means that no permission check will be executed.
alias	Specifies the unique identifier for this connector. This must be set to `defaultJmx` so that Software AG Platform Manager recognizes the connector.

When changing the port of the JMX connector, change the *port_number* in the filename to correspond to the new value of the `port` property in the file.

Important: Do not edit the com.softwareag.jmx.connector.pid-port_number.properties file unless Software AG Global Support asks you to make changes in the file.

Configuring the Java Service Wrapper

Software AG Runtime runs on the on the Software AG Common Platform, which in turn runs in a Java Virtual Machine (JVM). The Java Service Wrapper is an application developed by Tanuki Software, Ltd. It is a utility program that launches the JVM in which Software AG Runtime runs.

In addition to launching the JVM, the Java Service Wrapper offers features for monitoring the JVM, logging console output, and generating thread dumps. The following sections describe how Software AG Runtime uses the features of the Java Service Wrapper. For an overview of the Java Service Wrapper, see the webMethods cross-product document, *Working with the webMethods Product Suite and the Java Service Wrapper*.

For more information on how webMethods products use the Java Service Wrapper, see the documentation for the products you require.

The Java Service Wrapper Configuration Files

For Software AG Runtime, the configuration files for the Java Service Wrapper reside in the following directory:

Software AG_directory\profiles\CTP\configuration

When you start Software AG Runtime, property settings in the following files determine the configuration of the JVM and the behavior of the logging and monitoring features of the Java Service Wrapper.

File name	Description
wrapper.conf	Contains property settings that are installed by Software AG Runtime. *Do not modify the contents of this file unless asked to do so by Software AG.*
custom_wrapper.conf	Contains properties that modify the installed settings in wrapper.conf. If you need to modify the property settings for the Java Service Wrapper, you make your changes in this file.

The configuration file contains settings that enable you to control various properties. The properties are divided into categories, such as Environment variables, Wrapper Java Properties, and Logging properties.

The following sections describe configuration changes that Software AG Runtime supports for the Java Service Wrapper. Unless directed to do so by Software AG, do not make any configuration changes to the Java Service Wrapper other than the ones described here.

Configuring Wrapper Java Properties

You can use the `wrapper.java.additional` properties to define and configure additional Java properties that are passed to the Java Virtual Machine (JVM) when it is started. The properties that you define are applied to the JVM itself, not the wrapped application. Each additional property is defined in the following format:

```
wrapper.java.additional.<n>
```

where <n> starts from one and increases by one for each consecutive property. As a rule, the chain of additional properties must not be broken by missing property numbers.

When you run the Java wrapper on UNIX or Linux based operating systems, for each entry you can define the following flag: `wrapper.java.additional.<n>.stripquotes property = TRUE`. You use this flag to omit unclosed quotes if, for example, you use

references that contain spaces. If the flag is set, the Java wrapper replaces symbols using the following convention:

- \\ (double backlash) becomes \ (single backlash);
- \" (backlash and quotes) becomes "(quotes);
- "(quotes) becomes null (quotes are omitted).

All other symbols remain unchanged.

The wrapper.conf file is installed with several `wrapper.java.additional` property settings. If you need to modify an installed setting, copy the existing property from the wrapper.conf file to the custom_wrapper.conf file and edit the property in the custom_wrapper.conf file.

Note: Do not edit the property's sequence number. The modified property in the custom_wrapper.conf must have the same sequence number as in the wrapper.conf file.

If you need to apply more `wrapper.java.additional` properties to the JVM, specify the additional properties in the custom_wrapper.conf file and increment the `<n>` sequence for each new property you add. For example, if the wrapper.conf file contains `wrapper.java.additional.1` through `wrapper.java.additional.4`, add your new property as `wrapper.java.additional.5`.

Configuring Memory Properties

You can use the following properties to configure the JVM memory allocation. You can view the installed settings in wrapper.conf. If you need to modify an installed setting, copy the existing property from the wrapper.conf file to the custom_wrapper.conf file and edit the property in the custom_wrapper.conf file.

Property	Value
wrapper.java.initmemory	`256` (default). If you specify a value for this property, make sure it is smaller than the value of the `wrapper.java.maxmemory` property.
wrapper.java.maxmemory	`512` (default). Specifies the maximum amount of memory allocated by the JVM at startup.

Note: To use the default values that are configured in the JVM itself, set these properties to zero (0) in the custom_wrapper.conf file.

- If you configure the default JVM memory amount with a `0` value, you can set the –Xms parameter manually as an additional property in the configuration file.
- If you configure the memory amount in the Java wrapper with a non-zero value, the wrapper adds an appropriate –Xms parameter.

Configuring Wrapper Logging Properties

In this category, you can configure the properties that control the internal logging of the Java Service Wrapper, as well as the JVM-related output that is displayed in the console window when the Java Service Wrapper starts. These properties do not affect the logging within the Java application.

The table below describes the available properties in the category, the recommended values and the corresponding descriptions.

Property	Description
wrapper.console.format	Specifies the logging output format. Default value is `PM`. The default value is also used when you do not specify any tokens or if you disable the property. You can combine the tags without any restrictions, however, it is recommended to provide the `M` tag in the end.

Set to...	To specify...
`L`	Log level.
`P`	Whether the log is related to the Java Service Wrapper or JVM.
`D`	Thread.
`T`	Time.
`Z`	Time in milliseconds.
`U`	Approximate uptime in seconds. This value is based on the internal tick counter and applicable up to one year after startup.
`M`	Message.

Property	Description
wrapper.console.loglevel	Specifies the log level of the console output. Default value is `INFO`.

Set to...	To log...
`NONE`	No output.
`FATAL`	Only fatal error messages.
`ERROR`	All error messages (includes `FATAL`).
`WARN`	All warning messages (includes `ERROR` and `FATAL`).
`STATUS`	All state changes.
`INFO`	All JVM output and informative messages.
`DEBUG`	Detailed debug information.
`ADVICE`	Messages that facilitate problem resolutions. You can disable `ADVICE` messages by setting the `wrapper.adviser` property.
`NOTICE`	Messages that indicate license-related issues and appear when a trial license is used.

wrapper.logfile	Specifies the path to the log file on the file system where the console output is logged. If you do not configure the property, the log is saved to a default file called wrapper.log in the directory of the executable wrapper.
wrapper.logfile.format	Specifies the logging output format. The default value is `LPTM`. It is used even if you do not specify any tokens or if you disable the property. You can combine the tags without any restrictions, however, it is recommended to provide the `M` tag in the end.

Set to...	To specify...

Property	Description	
	`L`	Log level.
	`P`	Whether the log is related to the Java Service Wrapper or JVM.
	`D`	Thread.
	`T`	Time.
	`Z`	Time in milliseconds.
	`U`	Approximate uptime in seconds. This value is based on the internal tick counter and applicable up to one year after startup.
	`M`	Message.
wrapper.logfile.loglevel	Specifies the log level of the console output. The default value is `INFO`.	
	Set to...	**To log...**
	`NONE`	No output.
	`FATAL`	Only fatal error messages.
	`ERROR`	All error messages (includes `FATAL`).
	`WARN`	All warning messages (includes `ERROR` and `FATAL`).
	`STATUS`	All state changes.
	`INFO`	All JVM output and informative messages.
	`DEBUG`	Detailed debug information.
	`ADVICE`	Messages that facilitate problem resolutions. You can disable `ADVICE` messages by setting the `wrapper.adviser` property.

Property	Description
	`NOTICE` Messages that indicate license-related issues and appear when a trial license is used.
wrapper.syslog.loglevel	Specifies the log level of the console output. Default value is `NONE`.
	Set to... **To log...**
	`NONE` No output.
	`FATAL` Only fatal error messages.
	`ERROR` All error messages (includes `FATAL`).
	`WARN` All warning messages (includes `ERROR` and `FATAL`).
	`STATUS` All state changes.
	`INFO` All JVM output and informative messages.
	`DEBUG` Detailed debug information.
	`ADVICE` Messages that facilitate problem resolutions. You can disable `ADVICE` messages by setting the `wrapper.adviser` property.
	`NOTICE` Messages that indicate license-related issues and appear when a trial license is used.
wrapper.logfile.maxsize	If the logging file reaches a predefined critical size and if log file rollover is enabled, the logging content is transferred to another file on the file system. Use this property to specify the critical log file size in megabytes (m) or kilobytes (k). For example, 10m limits the file size to 10MB. To set unlimited file size, enter `0` as a property value.
wrapper.logfile.maxfiles	When logfile rollover is enabled, this property specifies the maximum number of rollover files. The

Property	Description
	current logging file is not considered in that number. To set an unlimited number of files, enter `0` as the property value.

Fault Monitoring

The Java Service Wrapper can monitor the JVM for the certain conditions and then restart the JVM or perform other actions when it detects these conditions.

The following table describes the fault-monitoring features Software AG Runtime uses or allows you to configure. To learn more about these features, see the webMethods cross-product document, *Working with the webMethods Product Suite and the Java Service Wrapper*.

Feature	Enabled?	User configurable?
JVM timeout	No	No. Do not modify the properties unless asked to do so by Software AG.
Deadlock detection	No	No. Do not enable this feature.
Console filtering	No	No. Do not enable this feature.

Generating a Thread Dump

The Java Service Wrapper provides a utility for generating a thread dump of the JVM when Software AG Runtime is running as a Windows service. A thread dump can help you locate thread contention issues that can cause thread blocks or deadlocks.

For information about generating a thread dump using the Java Wrapper Service, see the webMethods cross-product document, *Working with the webMethods Product Suite and the Java Service Wrapper*.

Configuring the Runtime Log Settings

Software AG Runtime delivers Journal Logging for logging purposes. To enable users to configure log settings, Software AG Runtime installation contains a log_config.xml, located in the *Software AG_directory*\profiles\CTP\configuration\logging directory.

Hot Configuration Update

Software AG Runtime runs a watchdog service that monitors the files under the *Software AG_directory*\profiles\CTP\configuration\com.softwareag.platform.config.propsloader directory and the JAAS configuration file at *Software AG_directory*\profiles\CTP\configuration\jaas.config and updates the runtime if changes to those files are detected. The watchdog maintains a set of configuration loaders for each supported file type (currently .properties and jaas.config). The poll interval for each configuration loader can be configured by modifying the *Software AG_directory*\profiles\CTP\configuration\com.softwareag.platform.config.propsloader\com.softwareag.config.watchdog.pid.properties file. The following properties can be configured:

Property	Description
`poll.default`	The poll interval in milliseconds to be used for configuration loaders that do not explicitly specify a poll interval. The default value is `5000`. When the value of this property is set to less than `1000`, polling is disabled for all loaders without values and all loaders with values set to less than `1000`.
`poll.file.property.loader`	The poll interval in milliseconds to be used for property file configuration loader. No default value. When the value of the property is set to less than `1000`, the value is reset to the value of the `poll.default` property.
`poll.jaas.file.loader`	The poll interval in milliseconds for the JAAS configuration file loader. The default value is `1000`. When the value of the property is set to less than `1000`, the value is reset to the value of the `poll.default` property.

Using Path Tokens

Software AG Runtime supports the usage of path tokens in the properties files under the *Software AG_directory*\profiles\CTP\configuration\com.softwareag.platform.config.propsloader directory and in the JAAS configuration file located at *Software AG_directory*\profiles\CTP\configuration\jaas.config.At runtime the path tokens are detected and replaced with the respective absolute location. These

replacements take place in memory only and the files on disk will always contain values with path tokens.

The following standard path tokens are supported:

- - osgi.install.area - resolves to *Software AG_directory*/profiles/<*profile*>/
- - osgi.instance.area - resolves to *Software AG_directory*/profiles/<*profile*>/workspace/
- - osgi.configuration.area - resolves to *Software AG_directory*/profiles/<*profile*>/configuration/
- - sag.install.area - resolves to *Software AG_directory*

To specify that a path token must be resolved to a regular path, add a `@path:` prefix to the token name. If the path token must be resolved to an URL, add a `@url:` prefix instead.

The following examples are valid for the jaas.config file:

For a property that contains the Software AG installation directory (C:/SoftwareAG) as an absolute path, for example, `someProperty=C\:/SoftwareAG/common/conf/someFile.conf` the property value can be modified by replacing `C\:/SoftwareAG` with `@path:sag.install.area`. After the change, the property will look like this: `someProperty=@path:sag.install.area/common/conf/someFile.conf`.

If the property contains an URL instead of an absolute path, for example, `someProperty=file\:\C\:/SoftwareAG/common/conf/someFile.conf` the property value can be modified replacing `file\:C\:/SoftwareAG` with `@url:sag.install.area`. After the change, the property will look like this: `someProperty=@url:sag.install.area/common/conf/someFile.conf`

Important: When working in a .properties file you should use the `\` symbol to escape symbols that may potentially break the configuration, for example, *:* .

3 Starting and Stopping Runtime

■ Starting and Stopping Runtime 28

Starting and Stopping Runtime

Your Software AG Runtime installation directory contains startup scripts which enable you to start and stop the Software AG Runtime instance. Different scripts are available depending on your operating system.

Note: Software AG Runtime uses the Tanuki Software, Ltd.Java Service Wrapper at startup. For more information about the Java Service Wrapper, see the official Tanuki documentation.

Starting and Stopping Runtime in Windows

The Software AG Runtime service is *Software AG Runtime <version>* . It is registered to start up automatically at system start. You can start, stop, and modify the service in **Control Panel** > **Administrative Tools** > **Services**.

Note: You can modify the startup type of the service in the **Services** window. The startup type of the service can be set to **Automatic**, **Manual**, or **Disabled**. The recommended startup type is **Automatic**.

Starting and Stopping Runtime in UNIX

Important: Before you start the daemon processes on UNIX, you need to set sufficient data user limits for the shell which starts the Software AG Runtime daemons. Having an insufficient data user limit might result in an OutOfMemoryError java exception at startup. For more information on setting data user limits, see the main page for *ulimit* or contact your system administrator.

The installation registers the daemons for Software AG Runtime in the UNIX init structure so that Software AG Runtime starts automatically when the system starts. The following scripts are installed:

For Linux and Solaris:

- /etc/init.d/sag<n>ctp98
- /etc/rc<R>.d/K20sag<n>ctp98
- /etc/rc<R>.d/S60sag<n>ctp98

For AIX:

- /etc/sag<n>ctp98
- Entry in /etc/inittab:

sag<n>ctp98:<R>:wait:/etc/sag<n>ctp98 start > /dev/console 2>&1

For HP-UX:

- /sbin/init.d/sag<n>ctp98
- /sbin/rc<R>.d/K20sag<n>ctp98
- /sbin/rc<R>.d/S60sag<n>ctp98

Note: In the script names <R> refers to the system runlevel and <n> refers to a number which gets incremented by 1 for each installation on the local machine. To temporarily deactivate a service, you need to remove or rename those files manually. Native configuration tools like the Yast Run-Level-Editor on Linux do not work.

The path to the Software AG Runtime daemon is <SAGROOT>/common/bin/wrapper-3.5.25 and the daemon can have several child processes.

To start and stop Runtime manually

- Start the Software AG Runtime daemon.

  <SAGROOT>/profiles/CTP/bin/startup.sh

- Stop the Software AG Runtime daemon.

  <SAGROOT>/profiles/CTP/bin/shutdown.sh

4 Securing Runtime

■ Managing the Runtime Security 32

Managing the Runtime Security

The Software AG Runtime security is managed by the jaas.config file located in the *Software AG_directory*\profiles\CTP\configuration directory. This security configuration file contains application contexts for the different parts of the Software AG Runtime authentication. You can use the default login modules in the file or you can add your own modules that enable the use of SSO. The default authentication mechanism checks the username and password against the local user repository handled by the InternalLoginModule. The local user repository is in the users.txt file located in the *Software AG_directory*/common/conf/ directory.

For more information about available authentication mechanisms, see the *Software AG Security Infrastructure Documentation*.

5 Completing the Software AG Runtime Installation

- Post-Installation Steps 34

Post-Installation Steps

After installing Software AG Runtime, it is recommended to complete the installation by doing the following steps:

1. Change the default Software AG Web Services Stack credentials. For more information on changing login credentials, see *Using Software AG Web Services Stack* .
2. Change the default credentials of the internal user repository. For more information on changing credentials of the internal user repository, see the *Software AG Security Infrastructure Documentation* .
3. Generate a Java keystore file with a key pair and respective certificate for the Tomcat HTTPS connector. For more information on how to configure the HTTPS connector, see "Working with HTTPS Connectors" on page 13.