Universal Messaging Installation Guide

Version 10.11

October 2021

This document applies to Software AG Universal Messaging 10.11 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.

Document ID: NUM-IG-1011-20230719

Table of Contents

About this Documentation

Online Information and Support

Software AG Documentation Website

You can find documentation on the Software AG Documentation website at https://documentation.softwareag.com.

Software AG Empower Product Support Website

If you do not yet have an account for Empower, send an email to empower@softwareag.com with your name, company, and company email address and request an account.

Once you have an account, you can open Support Incidents online via the eService section of Empower at https://empower.softwareag.com/.

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.

If you have any questions, you can find a local or toll-free number for your country in our Global Support Contact Directory at https://empower.softwareag.com/public_directory.aspx and give us a call.

Software AG Tech Community

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

- Access product documentation, if you have Tech Community 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.
- Go to our public GitHub and Docker repositories at https://github.com/softwareag and https://hub.docker.com/u/softwareag and discover additional Software AG resources.

Data Protection

Software AG products provide functionality with respect to processing of personal data according to the EU General Data Protection Regulation (GDPR). Where applicable, appropriate steps are documented in the respective administration documentation.

1 Installation Overview

This guide describes how to install and configure the Universal Messaging product.

The guide contains the following information:

- How to perform the installation procedure.
- How to perform various configuration and administration tasks after you have installed the product.
- How to use the Instance Manager to create instances of realm servers, the Enterprise Manager and template applications.

2 Installation using the Software AG Installer

The Software AG Installer is a generic tool for installing Software AG products.

Universal Messaging can be licensed as part of a product bundle, so when you run the Software AG Installer, a dialog allows you to select the appropriate bundle. A subsequent dialog shows the individual products that you can install from the bundle, including Universal Messaging.

Overview of the Documentation for using the Software AG Installer

For the installation using the Software AG Installer, refer to the following documents:

- **Using the Software AG Installer**

  This document describes how to use the Software AG Installer tool. The usage of the Software AG Installer is the same for all products, so the documentation of the Software AG Installer does not refer explicitly to Universal Messaging.

  To access the document *Using the Software AG Installer*, do the following:

  1. Log in to the Software AG documentation web site at http://documentation.softwareag.com/, using the Empower login ID and password that you have received by email when you licensed the product.
  2. Select the link for the Software AG Installer.
  3. The selected page lists several versions of the installer documentation, each shown with a release date. Select the version of the installer documentation that corresponds to the release date of the Software AG Installer you are using. The release date of the Software AG Installer is generally included in the file name of the downloaded executable file. You can also find the release date of the Software AG Installer by clicking the "About" link when you run the Software AG Installer.

- **Installing Software AG Products**

  This document contains specific installation information about many Software AG products, including Universal Messaging.

  The most recent version of the document *Installing Software AG Products* is available in the documentation web site using the following URL:

  http://documentation.softwareag.com/webmethods/Installing_Software_AG_Products.htm

- **Upgrading Software AG Products**

  If you are upgrading from a previous product version, refer also to this document. This document contains information about how to upgrade an existing Software AG product version to a new version.

  The most recent version of the document *Upgrading Software AG Products* is available in the documentation web site using the following URL:

  http://documentation.softwareag.com/webmethods/Upgrading_Software_AG_Products.htm

If you need to access versions of the documents *Installing Software AG Products* and *Upgrading Software AG Products* for previous product releases, proceed as follows:

1. From the starting page at http://documentation.softwareag.com/, follow the "webMethods" link.
2. Navigate to the webMethods Product Suite, then select the Suite version number that matches the Universal Messaging version number.
3. Select the link that deals with installation topics.

3 Starting the Realm Server

This section describes how to **start** the Universal Messaging realm server by using the command prompt of your operating system or by using operating system shortcuts. For similar instructions on how to **stop** the realm server, see the section "Stopping the Realm Server" on page 17.

Note:
Starting and stopping the realm server is also possible using the Command Central interface. Refer to the appropriate section of the *Administration Guide* for information.

Automatic startup of the realm server

If you selected the option to install the realm server as a service (on Windows) or a daemon (on UNIX-based systems) while you were installing the Universal Messaging product, the service/daemon will start automatically every time you boot your machine.

If you stop the service/daemon and wish to restart it without rebooting your machine, run the `nserver.bat` command (on Windows) or the `nserver` script (on UNIX-based systems). This command/script is located in the `bin` directory of your selected realm server.

Manual startup of the realm server

If you did not select the installation option to create the realm server as a service/daemon, you need to start the realm server manually.

Starting the realm server manually on Windows

There are several ways to start a realm server:

- Method 1: For Windows operating systems, the installation procedure for Universal Messaging installs a Start Menu entry called **Start Servers > Start Universal Messaging Realm Server > Start *<InstanceName>*** under the Universal Messaging entry, where *<InstanceName>* is the name of the realm server. When you select this shortcut, the realm server starts up.
- Method 2: You can open a console prompt by selecting the Start Menu entry **Tools > Universal Messaging Clients > *<InstanceName>* > Realm Server Command Prompt**. This opens a console prompt at the `bin` directory of your selected realm server. At the prompt, type `nserver`, and the realm server will start up.

Starting the realm server manually on UNIX-based systems

For UNIX-based operating systems, you can start the realm by running the `nserver` script in the `bin` directory of your selected realm server.

Avoiding server stop on user logoff

If the realm server is not configured as a daemon, `nserver` will start the realm server in console mode, which will cause the realm server to stop when the user who started it logs off.

This server stop resulting from a user logoff can be avoided by using the `nohup` command to start the realm server:

```
nohup ./<InstallDir>/UniversalMessaging/server/<InstanceName>/bin/nserver &
```

Forcing console mode startup

You can force the realm server to run in a console even if you installed it as a service/daemon. To do this, open a console prompt, change your working directory to the `<InstanceName>/bin` directory, and use the following command:

```
nserverdaemon -c
```

Behavior of the realm server during startup

To check that the realm server has started properly, examine the realm server log file `nirvana.log` in the location `<InstallDir>\UniversalMessaging\server\<InstanceName>\data`, where *<InstallDir>* is the disk root location of your Universal Messaging installation, and ensure there are no errors being reported. Also check the log file for a completion message such as the following:

```
[Tue Apr 24 10:59:22.374 CEST 2018] ... - Startup: Realm Server Startup sequence
completed
```

For more information about the contents of the log file, refer to the section *Universal Messaging Enterprise Manager : Logs Panel* in the *Administration Guide*.

Switching between Automatic startup and Manual startup

If you installed the realm server to start manually, but would now prefer it to start automatically as a service/daemon, you can *register* the realm server as a service/daemon. To do this, proceed as follows:

Changing from manual to automatic startup:

- **Windows:** Ensure that you are a user with Administration rights, then run the command `registerService.bat` located in the directory `<InstallDir>\UniversalMessaging\server\<InstanceName>\bin`.
- **UNIX:** Ensure that you are a user with sudo rights, then use the centralized `daemon.sh` script:

  ```
  <InstallDir>/common/bin/daemon.sh -f <Path>/nserverdaemon
  ```

where <*Path*> is `<InstallDir>/UniversalMessaging/server/<InstanceName>/bin`.

Changing from automatic to manual startup:

If you installed the realm server to start automatically as a service/daemon, but you want to change this setup so that you start the realm server manually, you can *deregister* the realm server as a service/daemon. To do this, proceed as follows:

- **Windows:** Ensure that you are a user with Administration rights, then run the command `unregisterService.bat` located in same directory as the `registerService.bat` command.
- **UNIX:** Ensure that you are a user with sudo rights, then use the centralized `daemon.sh` script:

  ```
  <InstallDir>/common/bin/daemon.sh -d <Path> -R
  ```

  where <*Path*> is `<InstallDir>/UniversalMessaging/server/<InstanceName>/bin`.

Getting the current status of the realm service/daemon

To display the current status of the service/daemon of the realm server, proceed as follows:

- **Windows:** Enter the following command at the command prompt:

  ```
  <InstallDir>\UniversalMessaging\server\<InstanceName>\bin\nserverdaemon -q
  ```

  Note:
  The command requires administrator rights and needs to be run from a console that has administrator rights.

- **UNIX:** To check if the realm server is installed as a daemon, enter the following command at the command prompt.

  ```
  <InstallDir>/common/bin/daemon.sh -d <Path> -l
  ```

  where <*InstallDir*> and <*Path*> are as defined above. You do not require sudo rights to run this command.

4 Stopping the Realm Server

This section describes how to **stop** the Universal Messaging realm server by using the command prompt of your operating system or by using operating system shortcuts. For similar instructions on how to **start** the realm server, see the section "Starting the Realm Server" on page 13.

Note:
Starting and stopping the realm server is also possible using the Command Central interface. Refer to the appropriate section of the *Administration Guide* for information.

Stopping the realm server on Windows

For Windows operating systems, the installation procedure for Universal Messaging installs a Start Menu entry called **Stop Servers > Stop Universal Messaging Realm Server > Stop *<InstanceName>***, where <*InstanceName*> is the name of the realm server. Select this entry to stop the realm server.

Alternatively, you can open a command prompt, then set the working directory to the `bin` directory of your selected realm server, then enter the command `nstopserver.bat`.

Note:
The `nstopserver.bat` command is a non-blocking command, which means it will initiate a realm server shutdown and will exit, without waiting for the realm server process to terminate. To ensure that the realm server process has terminated, check the server log file, as described below in the section "Realm Server behavior while shutting down" on page 17.

Stopping the realm server on UNIX-based systems

For UNIX-based operating systems, stop the realm server by opening a command prompt, then set the working directory to the `bin` directory of your selected realm server, then run the script `nstopserver`.

Note:
As described above for Windows, the `nstopserver` script is a non-blocking script.

Realm Server behavior while shutting down

While the realm server is shutting down, it closes down all client links and resources. This may take some time, due to the operating system resources that the realm server uses. The realm server

will automatically perform a complete shutdown within 10 seconds if the graceful shutdown has not yet completed, at which point the realm server will perform an immediate shutdown. The realm server will generate a thread dump which will be written to the log file prior to shutdown.

The realm server will log the shutdown in the server log file `nirvana.log`. A successful shutdown is indicated by entries similar to the following:

```
Shutdown: Realm Server completed shutdown sequence, process existing normally
... (metadata related to the session that has just terminated) ...
--------- Log File Closed ---------
```

5 Post-Installation Procedures

The information in the following sections describes procedures that you can use after the installation has completed.

Server Memory Modes

Server Memory Modes

The performance and behaviour of the Realm Server is inseparably linked to the amount of maximum heap memory allocated to the Java VM hosting it. The Realm Server is capable of scaling depending on the hardware platform it is hosted on, and that is determined by the memory available to the Java VM. The Realm detects this and switches its mode of operation to *Small Memory Mode, Medium Memory Mode* or *Large Memory Mode*.

Small Memory Mode

Allocating 16MB or less of heap memory to the Java VM hosting the Universal Messaging Realm will make it operate in small memory mode. This is confirmed at start-up by a log entry like the following:

```
Audit,Setting Server mode to Small Memory Mode
```

The Universal Messaging Realm small memory mode should be used when running a Realm on mobile or embedded devices, or other machines with very limited memory resources available. Apart from limited memory available to store events in reliable channels, all thread pooled sub systems are changed to have only one thread. It is therefore recommended that persistent channels should always be used on such Realms. The performance will also be reduced by the fact that all caching is disabled in this mode.

All of the functionality provided by the innovative Universal Messaging server side Realms are available in the small memory mode and hence on handheld devices etc.

Medium Memory Mode

Allocating 16MB or more of heap memory to the Java VM hosting the Universal Messaging Realm will make it operate in medium memory mode. This is confirmed at start-up by a log entry like the following:

```
Audit,Setting Server mode to Medium Memory Mode
```

The Universal Messaging Realm medium memory mode should be used when running Realms on development or where memory is at a premium. All thread pooled sub systems will start up with our recommended default values for this mode. Tuning the Realm to higher values for those sub systems will increase the Realm's memory requirements and increase caching age values.

Large Memory Mode

Allocating 70MB or more of heap memory to the Java VM hosting the Universal Messaging Realm will make it operate in large memory mode. This is confirmed at start-up by a log entry like the following:

```
Audit,Setting Server mode to Large Memory Mode
```

The Universal Messaging Realm large memory mode should be used when running Realms on development, staging or production environments, or when using reliable channels. All thread pooled sub systems will start up with our recommended default values for this mode.. Tuning the Realm to higher values for those sub systems will increase the Realm's memory requirements and increase caching age values.

JMS Configuration

The Universal Messaging Realm server is designed to automatically support applications that use the provided Universal Messaging Provider for JMS. Such applications however need a valid JNDI context configuration in order to remain vendor neutral.

Universal Messaging features a JNDI service provider that can operate using any of the Universal Messaging transport protocols (nsp, nhp, nsps and nhps) as well as tools that allow configuration of the latter or any JNDI context is required from LDAP to NIS. The Universal Messaging JNDI provider uses a channel called /naming/defaultContext to store JMS references and the implementation class is com.pcbsys.nirvana.nSpace.NirvanaContextFactory.

JNDI configuration can be performed in 2 ways. The first is by using a command line application (with full source code provided) called JMSAdmin. For more information about how to use this application please check the appropriate developer guide section. The second is by using the realm JNDI configuration panel in the Universal Messaging Enterprise Manager.

On the client side, you would need to set the standard java.naming.factory.initial key to the aforementioned com.pcbsys.nirvana.nSpace.NirvanaContextFactory and pass the JNDI provider URL of the realm under the nirvana.provider.url key or the standard java.naming.provider.url key. Note that the JNDI API suggests two ways of defining these properties - either by setting them as system properties, or by passing them in a JDNI environment Hashtable argument. When the NirvanaContextFactory is searching for the provider URL, it will by default first check the system properties, and if not found, it will consult the environment Hashtable argument. To reverse the order of lookup, you can set a system property key nirvana.provider.urlpref.sysprops to the value "N".

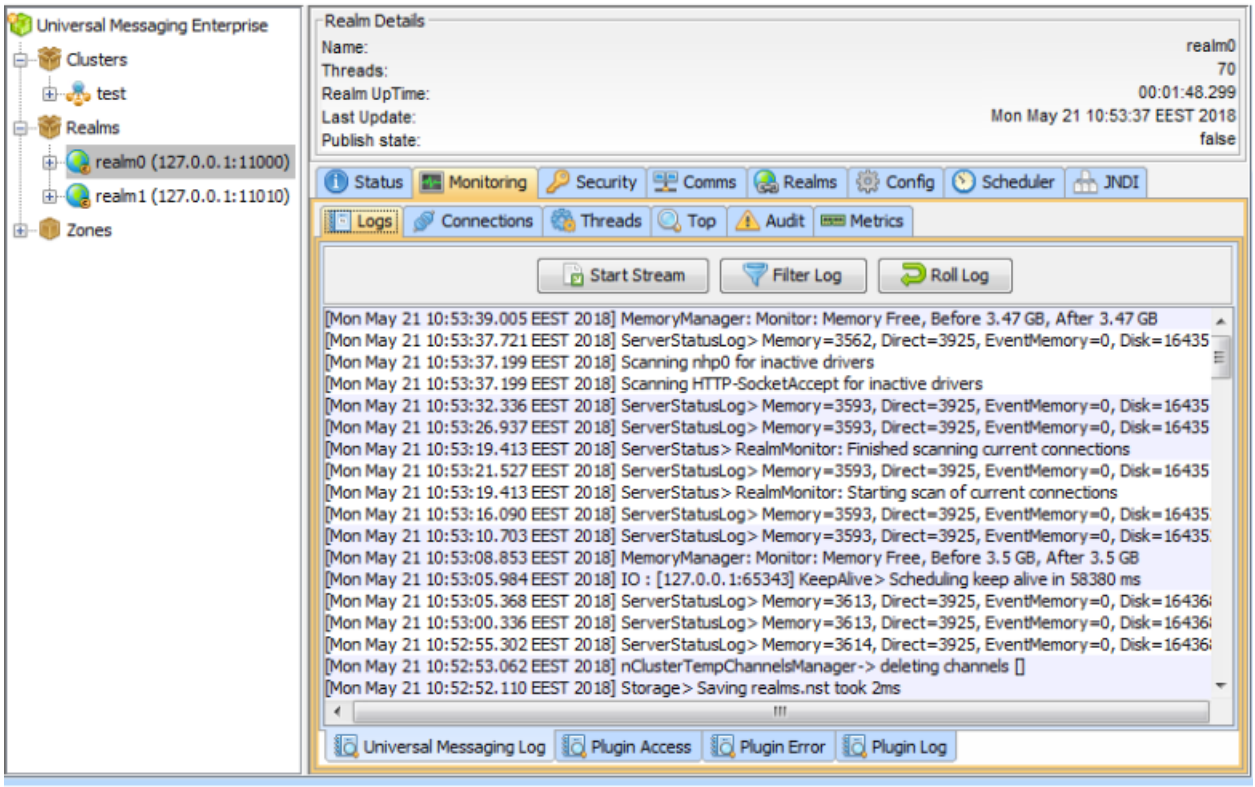
How to access the Universal Messaging log file

The Universal Messaging log file can be accessed using the Universal Messaging Enterprise Manager GUI by selecting the **Monitoring** tab and then the **Logs** tab. This provides access to the Universal Messaging log file itself as well as the log files associated with the Universal Messaging Realm Plugins. To switch between the available log files select the appropriate tab from the bottom of the logs panel.

See the description of the **Logs** panel in the documentation of the Enterprise Manager for more details about the log file.

Universal Messaging log files can also be accessed programmatically using the Universal Messaging Admin API.

If multiple realms have been added to the namespace then the log file for each can be accessed by clicking on that realm in the namespace and then selecting the **Monitoring** tab which will by default display the log panel.



The Dump file for Out-of-Memory Errors (OOME)

Universal Messaging automatically generates a heap dump file when an `OutOfMemoryError` (OOME) occurs in the JVM. The default directory where heap dump files are generated is:

```
<InstallDir>/UniversalMessaging/server/<InstanceName>/heap_dumps
```

where *<InstallDir>* is the disk location where the product is installed and *<InstanceName>* is the name of the Universal Messaging server instance that is running in the JVM.

In addition, when Universal Messaging is using the Azul Zulu JVM (which is included in the Universal Messaging distribution kit on the Windows, Linux and Solaris platforms), or an Oracle HotSpot JVM, the generated heap dump file will be compressed in a zip archive and older heap dump files will be deleted, preserving only the 10 most recent heap dumps files.

For JVMs of other suppliers, heap dumps are stored as is (no compression and no cleanup of older dump files).

Changing the dump file location

If you wish to configure a different heap dump directory than the default, proceed as follows:

1. Using a text editor, open the file `Server_Common.conf` that is located in *<InstallDir>*`/UniversalMessaging/server/`*<InstanceName>*`/bin`.

2. Adapt the `HEAP_DUMP_DIR` environment variable.

   You can specify an absolute path or a relative path. If you specify a relative path, the value must be relative to the location of the `Server_Common.conf` file.

   When using the Oracle HotSpot JVM, the configured location must already exist, since it is not created automatically.

The changes take effect at the next restart of the server instance.

How to Administer a Remote Universal Messaging Realm

A typical development setup involves installing a Universal Messaging Realm in a remote development server as well as the developer's workstation. This guide will help you connect to the remote development Universal Messaging realm for administration purposes.

A Universal Messaging realm by default enforces a security model that allows only the username running the server process to connect to it with full privileges, and that can only be done when connecting from localhost (127.0.0.1). Therefore, in order to be able to connect remotely you need to add an ACL entry for the user and the IP address you will connecting from. To do this you need to use the naddrealmacl command line tool on the development server as follows:

Windows Server Steps

1. Open a client command prompt from the Start Menu shortcut.

   For related information see the section *Running the Sample Applications* in the *Developer Guide*.

2. Type "naddrealmacl <user> <ip> full", where <user> is the OS username that the development workstation will use to connect and <ip> is the ip address of the development workstation. In this instance we specify that full access should be given to this user.

Linux / Solaris Server Steps

1. Open a console window (shell)

2. Type "cd <install dir>/client/<realm_name>/bin", where <install_dir> is your installation path and <realm_name> is the name you assigned to the realm during installation.

3. Type "export RNAME=nsp://localhost:9000", this sets an environment variable called `RNAME` that all samples and command line tools use in order to know how to connect to the server. In this instance we are using the Universal Messaging Socket Protocol on localhost and port 9000. If you have chosen a different port please adjust accordingly.

4. Type " ./naddrealmacl <user> <ip> full", where <user> is the OS username that the development workstation will use to connect and <ip> is the ip address of the workstation. In this instance we specify that full access should be given to this user.

Development Workstation Steps

1. Run your enterprise manager on the development workstation and click on the Connections menu, selecting Connect To Realm.
2. A dialog will pop up asking you to specify the RNAME to use. Similarly to what we did for the command line tool, we specify "nsp://<server ip>:9000", where <server ip> is the IP address where the server is running and 9000 is the port the server is listening on. Again if you have chosen a different port please adjust accordingly
3. Click ok and you should see your realm appear in the tree under the Realms folder.

License Management

Universal Messaging provides two types of license:

- A cost-free trial license for evaluation purposes.
- A purchasable commercial license that enables all product features or a tailored subset of features.

Note:
Universal Messaging ships with a trial license, which allows the server to run for a maximum of 90 days from first run.

If you install the product using the Software AG Installer, the Installer offers you the choice of using the trial license or activating the commercial license if you have purchased one. If you install with the trial license, you can activate the full license at a later time, as described below.

Upgrading from a trial license to a commercial license

If you installed the product with a trial license for testing or evaluation purposes, the installed trial license is called `licence.xml`. If you now wish to purchase a commercial license for production purposes, please contact us, and you will receive a commercial license file which is also called `licence.xml`.

To upgrade from the trial license to the commercial license, proceed as follows:

1. Make a backup copy of the trial license file.
2. Replace your existing trial `licence.xml` by the commercial `licence.xml`.
3. Restart your server.

There is no requirement to install any additional software.

The location of the license file is by default as follows:

`<InstallDir>/`UniversalMessaging`/server/<InstanceName>`

where <*InstallDir*> is the disk location where the product is installed, and <*InstanceName*> is the name of the realm server to which the license applies.

Upgrading an installed commercial license

Depending on the commercial license you have purchased, certain product features may be enabled or disabled. If you wish to enable additional product features that are not covered by the current commercial license, you can purchase an upgraded license that allows you to activate the additional features. To activate the upgraded license, replace the existing commercial license file `licence.xml` by the new license file, following the procedure described in the section "Upgrading from a trial license to a commercial license" on page 24 above. There is no need to install any new software.

Downgrading an installed commercial license

If for any reason you decide that your current commercial license enables more product features than you require, and you wish to switch to a more restrictive set of features, you can purchase a downgraded license that allows you to deactivate some product features.

To activate the downgraded license, replace the existing commercial license file `licence.xml` by the new license file, then restart the server. There is no need to install any new software.

Note:

NOTE: If you downgrade the license file, this may disable some product features that you rely on for production work, which may in turn may cause the server to stop working, or work in an inconsistent state. Therefore, if you plan to downgrade a license, we recommend you to ensure that the already configured server does not use features that are unavailable in the downgraded license. Additionally, we recommend you to test your downgraded environment before going into production.

Downgrading the license of a blank (freshly installed) server instance does not have these limitations.

License activation

If you change a product license file as described in the scenarios above, you need to restart the Universal Messaging server or cluster in order to activate the new license.

6 Universal Messaging Instance Manager

During the installation of Universal Messaging, you have the option of creating a default instance (called `umserver` by default) for all the components installed. If you need to create additional instances, this can be done using the `ninstancemanager` command line tool, which can be found under `<InstallDir>`/UniversalMessaging/`tools/InstanceManager/`.

Components

The `ninstancemanager` tool can create instances of the following component types:

- realm server (RS)
- Enterprise Manager (EM)
- template applications (TA)

In order to create an instance of a component, this needs to have been installed first.

Usage Message

Executing the `ninstancemanager` tool without any arguments provides a usage message as follows:

```
ninstancemanager <Action> <InstanceName> <Component> <Host> <Port> [DataDirectory]
```

- <Action> can be either `create`, `delete`, `query`, `deleteAll` (followed directly by a component), or `configure`.
- <*InstanceName*> can be any instance name.
- <Component> is the component the action applies on, namely `RS` (for Realm Server), `EM` (for Enterprise Manager), `TA` (for Template Applications) or `ALL` (for everything installed).
- <Host> is the hostname or IP that the template applications and Enterprise Manager will point to, and the adapter the realm will bind to.

  You can use the hostname instead of the IP when you wish to provide an environment that is not specific to the underlying IP address of the server. This will allow the UM server to be accessed only by its hostname, so if the IP address changes, the server will still be accessible.
- <Port> is the TCP port that the template applications and Enterprise Manager will point to, and the adapter the realm will bind to.

- <DataDirectory> is the realm server working directory. This parameter is optional, and the default value is <InstallDir>"/"UniversalMessaging"/server/"<InstanceName>.

Example 1: To create a new instance called `umserver2` and listening to all IPs on port `9001`, you would run:

```
ninstancemanager create umserver2 all 0.0.0.0 9001
```

Example 2: To create a new EM instance called `umserver2` and pointing to a realm on `192.168.1.100` port `9001` you would run:

```
ninstancemanager create umserver2 all 192.168.1.100 9001
```

Example 3: To delete all instances called `umserver2` you would run:

```
ninstancemanager delete umserver2 all
```

Example 4: To delete an EM instance called `umserver2` you would run:

```
ninstancemanager delete umserver2 em
```

Example 5: To query installed instances you would run:

```
ninstancemanager query
```

Example 6: To delete all realm server instances you would run:

```
ninstancemanager deleteAll rs
```

Querying Installed Instances

Running the `ninstancemanager` tool with the query action displays a list of currently installed instances. For example:

```
ninstancemanager query
```

will display an output similar to the following in a default installation (taking release 9.8.0 as an example):

```
Universal Messaging installation query
--------------------------------------
Realm Server Installed Version:
e2ei/11/NUM_9.8.0.0.13321/UniversalMessaging/NUMRealmServer
Instances: umserver
Enterprise Manager Installed Version:
e2ei/11/NUM_9.8.0.0.13321/UniversalMessaging/NUMEnterpriseManager
Instances: umserver
Template Applications Installed Version:
e2ei/11/NUM_9.8.0.0.13321/UniversalMessaging/NUMTemplateApplications
Instances: umserver
```

Creating Instances

Running the `ninstancemanager` tool with the create action allows you to create instances of all the installed components or a subset. In order to create an instance, you need to run the `ninstancemanager` as follows:

```
ninstancemanager create <InstanceName> <Component> <Host> <Port>
```

Where:

- *<InstanceName>* is a logical name for the instance which needs to be unique for each installation.
- <Component> is the component you wish to create an instance of. The possible values are `ALL` (for all components installed), `RS` (for a realm server instance), `TA` (for template applications instance) or `EM` (for Enterprise Manager instance).

Note:
There are some naming rules for instances of realm servers. See the section "Naming Rules for Realm Server Instances" on page 32 below for details.

Note that after you create a server instance you cannot change the server instance name or the realm name. When creating the server instance, the server instance name is set to the realm name by default.

Example: If we wanted to create an instance of all components installed called `testinstance`, bound to all IPs of the machine and listening on port `9002` you would enter:

```
ninstancemanager create testinstance all 0.0.0.0 9002
```

Output:

```
Created RS instance testinstance
Created TA instance testinstance
Created EM instance testinstance
```

You can then verify the instance's presence by issuing a query action:

```
ninstancemanager query
```

Output:

```
Universal Messaging installation query
--------------------------------------
Realm Server Installed Version:
e2ei/11/NUM_9.8.0.0.13321/UniversalMessaging/NUMRealmServer
Instances: testinstance , umserver
Enterprise Manager Installed Version:
e2ei/11/NUM_9.8.0.0.13321/UniversalMessaging/NUMEnterpriseManager
Instances: testinstance , umserver
Template Applications Installed Version:
e2ei/11/NUM_9.8.0.0.13321/UniversalMessaging/NUMTemplateApplications
Instances: testinstance , umserver
```

Deleting Instances

The `ninstancemanager` tool can be used to delete any instances created, including the default instance created using the installer. The components specified allow you to remove an instance for one component while keeping it for the others.

In order to delete an instance, you need to run the `ninstancemanager` as follows:

```
ninstancemanager delete <InstanceName> <Component>
```

Where:

- <*InstanceName*> is a logical name for the instance which needs to be unique for each installation.
- `<Component>` is the component you wish to create an instance of. The possible values are `ALL` (for all components installed), `RS` (for a realm server instance), `TA` (for template applications instance) or `EM` (for Enterprise Manager instance).

Example: If we wanted to delete a previously created instance of all components called `testinstance`, you would enter:

```
ninstancemanager delete testinstance all
```

Output:

```
RS instance testinstance has been deleted
TA instance testinstance has been deleted
EM instance testinstance has been deleted
```

You can then verify the instance's presence by issuing a query action:

```
ninstancemanager query
```

Output:

```
Universal Messaging installation query
--------------------------------------
Realm Server Installed Version:
e2ei/11/NUM_9.8.0.0.13321/UniversalMessaging/NUMRealmServer
Instances: umserver
Enterprise Manager Installed Version:
e2ei/11/NUM_9.8.0.0.13321/UniversalMessaging/NUMEnterpriseManager
Instances: umserver
Template Applications Installed Version:
e2ei/11/NUM_9.8.0.0.13321/UniversalMessaging/NUMTemplateApplications
Instances: umserver
```

Deleting All Instances of a Component

You can delete all instances of a component (`RS`, `EM`, `TA` or `ALL`) by using the `deleteAll` action and passing the component:

Example: If we wanted to delete all previously created instances of the component type `TA` (for template applications), you would enter:

```
ninstancemanager deleteAll ta
```

Output:

```
TA instance umserver has been deleted
```

You can then verify that the instance(s) have been deleted by issuing a query action:

```
ninstancemanager query
```

Output:

```
Universal Messaging installation query
--------------------------------------
Realm Server Installed Version:
e2ei/11/NUM_9.8.0.0.13321/UniversalMessaging/NUMRealmServer
Instances: umserver
Enterprise Manager Installed Version:
e2ei/11/NUM_9.8.0.0.13321/UniversalMessaging/NUMEnterpriseManager
Instances: umserver
Template Applications Installed Version:
e2ei/11/NUM_9.8.0.0.13321/UniversalMessaging/NUMTemplateApplications
Instances:
```

Configuring an Existing Server Instance

The `ninstancemanager` command line tool can be used to configure existing server instances. The tool provides two ways to import a configuration for an instance:

- Import a configuration from a predefined profile. These predefined profiles are shipped with the installation. The configure command allows you to view the available predefined profiles.
- Import a custom configuration XML file that you have previously exported. This option can be used to apply a specific configuration, unique for a separate customer use case.

Note:

- In both cases, before the import is performed, a backup of the existing configuration will be made. Backups will be created under <*InstallDir*>/UniversalMessaging/server/<*InstanceName*>/data/configBackup, where <*InstanceName*> is the name of your server instance.
- The server instance needs to be stopped when importing a configuration, otherwise an error message will be printed and the configuration will not be imported.

You can use the `ninstancemanager` tool to perform the following actions:

- Display the command's help text

  ```
  ninstancemanager configure
  ```

- List information about the available predefined profiles

  ```
  ninstancemanager  configure  --listProfiles
  ```

  The command lists information about the available predefined profiles. A short description about each of the profiles will be also printed.

  Currently two predefined profiles are available: `wM` (for webMethods) and `TC` (for Terracotta). Each profile contains a set of Universal Messaging server configuration values that fits best to the profile's use case.

Profile name	Description
wM	webMethods suite use cases (large messages, transactions, persistence)

Profile name	Description
TC	Standalone use cases (small messages, non-transactional data)

- Import a predefined profile

  You can import a predefined configuration profile. Use the `--listProfiles` command as shown above to list the available predefined profiles.

  The command to import a predefined profile is as follows:

  ```
  ninstancemanager configure umserver --importProfile=<ProfileName>
  --dataDir=<InstallDir>/UniversalMessaging/server/<InstanceName>
  ```

  Here, <ProfileName> is the name of the profile you wish to import.

  The parameter <*InstanceName*> is the name of the server instance.

  The parameter `--dataDir` is optional. If you do not supply this parameter, the default base data folder is assumed, which is <*InstallDir*>/UniversalMessaging/server/umserver.

- Import a custom configuration file

  You can import a custom configuration file using a command of the following form (note the use of the parameter `--import` rather than `--importProfile`):

  ```
  ninstancemanager configure umserver --import=<customConfigFile>
  --dataDir=<InstallDir>/UniversalMessaging/server/<InstanceName>
  ```

  For example:

  ```
  ninstancemanager configure umserver --import=c:\myConfig\myCustomConfig.xml
  --dataDir=<InstallDir>/UniversalMessaging/server/umserver
  ```

- Export a configuration to a file

  You can export the current configuration into an external file, using a command of the form:

  ```
  ninstancemanager  configure <InstanceName>  --export=<path>
  ```

  For example:

  ```
  ninstancemanager  configure umserver  --export=c:\myBackup\myBackup.xml
  ```

  The command will export the current configuration in a file, which afterwards can be imported with the `--import` command.

  Note:
  The export command can be run even if the server instance is running.

Naming Rules for Realm Server Instances

When you are creating a new realm server instance, some rules relating to the instance name apply.

A realm server instance name can be the same as the name of an Enterprise Manager instance or a Template Applications instance. Apart from this, the following restrictions apply:

- Realm server instance names used throughout your Universal Messaging infrastructure must be unique.
- Realm server instance names must not be the same as the name of any host, cluster, or any other non-realm component in your Universal Messaging infrastructure.
- Instance names can contain only lower case Latin letters (a-z), digits (0-9), an underscore, and a non-leading ASCII hyphen (minus) character.
- The length of instance names with a shortcut path string must be fewer than 258 characters.
- The length of instance names with an instance path string must be fewer than 258 characters.

The reason for the naming restrictions is that Universal Messaging uses the name of the realm server instance internally as a unique identifier.

7 Using Containers

Docker Scripts on GitHub

If you want to build your own Universal Messaging container image to run in a Docker container, there are scripts available on GitHub that provide a template for setting this up. The scripts and accompanying documentation can be used as provided or can be altered to take account of specific environment requirements. Universal Messaging system requirements have been updated to reflect that Docker with a CentOS image is now supported in line with the other Software AG products. The samples use CentOS 8 as a base image.

The scripts are available at https://github.com/SoftwareAG/universalmessaging-server-docker-samples.

Universal Messaging Container Images on Software AG Container Repository

As an alternative to building your own Universal Messaging container image, there are official Universal Messaging container images on the Software AG Container Repository. These are container images that Software AG builds, tests, and promotes on a regular basis. The 10.11 images use ReadHat UBI8 as a base image.

The available container images are:

- **Software AG Universal Messaging Server**, available at https://containers.softwareag.com/products/universalmessaging-server.

  This image contains the Universal Messaging server, and also the administration and monitoring tools available in the **Software AG Universal Messaging Tools** image.

- **Software AG Universal Messaging Tools**, available at https://containers.softwareag.com/products/universalmessaging-tools.

  This image contains just the tools for the command-line administration and monitoring of Universal Messaging. This image may be used in conjunction with the Universal Messaging Server image.

  The tools are described in the section *Syntax reference for command line tools* in the *Administration Guide*.

Creating Universal Messaging Container Images Using Software AG Installer

You can also build Docker images for Universal Messaging on Linux systems using Software AG Installer. This approach allows only limited customization, such as providing a custom base image. The images that you build using the Installer client are similar to the official Universal Messaging images on Docker Hub and can be used in the same manner.

For more information about building Docker images for Universal Messaging using Software AG Installer, see the documentation about creating Docker images on Linux in the *Using Software AG Installer* guide.

Running Universal Messaging Docker Images

After you build or pull a Universal Messaging Docker image, you can run that image by using the `docker run` command. You must map the default container port `9000` to one of the host ports, for example:

```
docker run -d -p 9000:9000 --name umservercontainer universalmessaging-server:10.11
```

You can then use the host ports to access the server remotely via a URL, for example:

```
nsp://dockerhostname:hostport
```

Optionally, you can specify the follwing environment variables when running an image:

- `REALM_NAME` - the name of the Universal Messaging realm.
- `INIT_JAVA_MEM_SIZE` - the initial Java heap size in megabytes.
- `MAX_JAVA_MEM_SIZE` - the maximum Java heap size in megabytes.
- `MAX_DIRECT_MEM_SIZE` - the maximum direct memory size in gigabytes.
- `BASIC_AUTH_ENABLE` - enables basic authentication on the server.
- `BASIC_AUTH_MANDATORY` - enables and manages basic authentication on the server.
- `STARTUP_COMMAND` - configues the server to execute a given command at startup.
- `LOG_FRAMEWORK` - enables the Log4j2 logging framework. The valid value is `log4j2`. If this variable is not specified, the server uses the default logging framework (`fLogger`).

The following is an example of running an image with the environment variables specified:

```
docker run -d -e REALM_NAME=umtest -e INIT_JAVA_MEM_SIZE=2048 -e MAX_JAVA_MEM_SIZE=2048

-e MAX_DIRECT_MEM_SIZE=3G -e BASIC_AUTH_ENABLE=Y -e BASIC_AUTH_MANDATORY=Y -e
LOG_FRAMEWORK=log4j2
-e STARTUP_COMMAND="runUMTool.sh CreateChannel -channelname=test
-rname=nsp://localhost:9000"
-p 9000:9000 --name umservercontainer universalmessaging-server:10.11
```

When Universal Messaging uses the default logging framework, the output is recorded in the nirvana.log and UMRealmservice.log files and is also displayed in the console. Each log entry

starts with the specific log file name with which that entry is associated. You can also fetch the logs of a continer by using the `docker logs umservercontainer` command.

When Universal Messaging uses Log4j2 as the logging framework, the server logs are directly displayed in the console and written in the UMRealmService.log file. The log entries are not prefixed by the log file name, because Universal Messaging generates only one log file.

Using the Log4J2 Logging Framework

Universal Messaging supports the Log4j2 logging framework for Docker images and provides a default Log4j2 configuration. To enable Log4j2 for Docker images, set the `LOG_FRAMEWORK=log4j2` environment variable as part of the `docker run` command.

The log4j2.xml default configuration file is located inside the Universal Messaging container under `/opt/softwareag/UniversalMessaging/lib/classes`. The default configuration does not store the nirvana.log log file on disk but displays the log output to STDOUT. The output of the UMRealmService.log file is displayed in the console.

If you use a custom log4j2.xml configuration, you must retain the `packages` configuration entry that lists the Universal Messaging extension package for Log4j2:

```
Configuration packages="com.softwareag.um.extensions.logger.log4j2"
```

To configure the server logging level, use the nAdminAPI or the `fLoggerLevel` logging configuration property in the Enterprise Manager. To configure that property, you must first set `LogLevelOverride` to `true`. A `false` value for `LogLevelOverride` restores the configuration from log4j2.xml and overrides `fLoggerLevel`.

The following functionalities are not supported on a Universal Messaging server using Log4j2:

- `DefaultLogSize` logging configuration property
- `LogManager` logging configuration property
- `RolledLogFileDepth` logging configuration property
- Dynamic reconfiguration using the `monitorInterval` Log4J configuration attribute
- Rolling the log file