Developing Microservices with webMethods Microservices Container

Innovation Release

Version 10.0

May 2017

WEBMETHODS

This document applies to webMethods Microservices Container Version 10.0 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: WMIC-DG-100-20170515

Table of Contents

About this Guide

This guide provides information about administering webMethods Microservices Container.

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 Getting Started with webMethods Microservices Container

What Are Microservices?

Microservices are independently deployable units of logic in which each microservice performs a single business function. Applications built in the microservices architectural style are developed as a suite of microservices.

Microservices can be implemented in various ways, including as a set of services or as event and channel definitions. Microservices can be distinguished from other types of services in the webMethods suite as follows:

Type of Service	Description
Integration Server service	Single function call, such as an operation on an interface or as part of an API (that is, a related set of Integration Server services).
Web service	Formal, WSDL-based grouping of operations, usable for SOA. Many of these can be hosted in a single runtime.
Microservice	Collection of operations implemented as services or as messages and message handlers, deployed in one or more related Integration Server packages.

What Is webMethods Microservices Container?

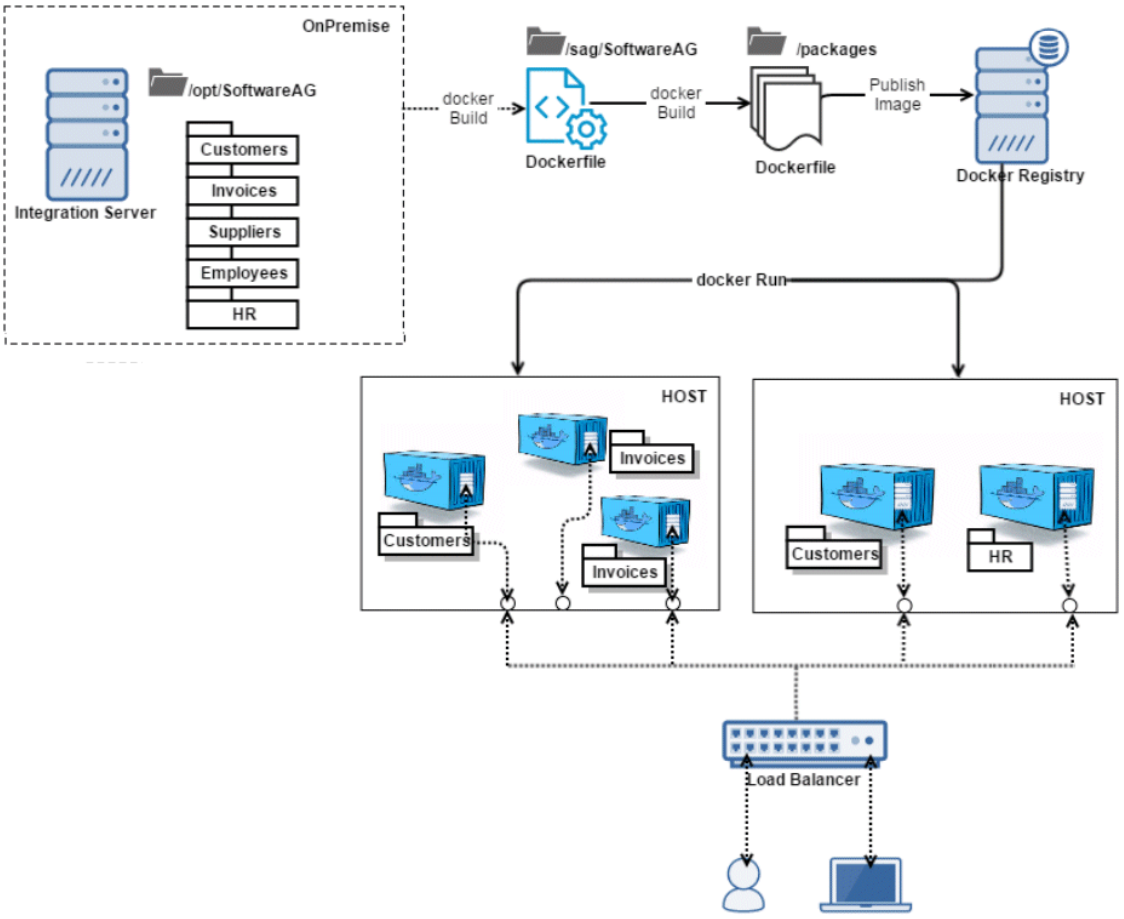
Software AG offers a lightweight container called webMethods Microservices Container to host microservices you develop in Software AG Designer. Using Microservices Container, you can deliver microservices as an Integration Server package that includes a set of related services, interfaces, document types, and triggers that subscribe to topics or queues, or as a set of related packages of this kind (for example, five packages relating to Human Resources functions).

Each microservice can run in its own Microservices Container and can communicate with lightweight mechanisms such as an HTTP resource API. However, you can also execute multiple microservices in the same Microservices Container. This hybrid solution enables you to separate microservices when needed, but also to group them when necessary. For example, suppose that you have two microservices that need to be scaled together in similar ways (that is, when you need a new instance of one, you need a new instance of the other). If you discover that one microservice is more heavily loaded than the other, or needs to be enhanced or updated more often, you could deploy the two microservices to separate Microservices Containers. If both microservices tend to be updated at the same time, you could cohost them in the same Microservices Container.

Microservices Container is fully compatible with webMethods Integration Server and can also host services you develop using Software AG Designer and Integration Server. While Microservices Container is optimized to have a reduced disk and memory footprint, you can convert it into a full Integration Server by installing additional modules, such as support for an external database.

Microservices Container provides out-of-the-box support for dynamic lookup of service endpoints using the open-source service registry named Consul. You can make your microservice available for remote access by registering the endpoint of the microservice container instance in the Consul service registry. Microservices Container provides services for automating the registration and deregistration process. Microservices Container provides facilities to look up the endpoint information which can be used to call the microservice at run time. You can also create your own package integrating with any other service registry provider. The package would provide services similar to those described for Consul in this guide.

Microservices Container is optimized for execution in a Docker container. You can run a microservice or a set of related microservices in a Docker container, obviating the need to purchase expensive virtual machines. Docker images include configuration, enabling you to deploy the exact same configuration anywhere. The Docker image can include one package or a set of related packages.



What Is Microservices Container Administrator?

Microservices Container Administrator is an HTML-based utility you use to administer Microservices Container. It allows you to monitor server activity, manage user accounts, make performance adjustments, and set operating parameters.

You can run the Microservices Container Administrator from any browser-equipped workstation on your network. Microservices Container Administrator is a browser-based application that uses services to accomplish its work.

2 Starting, Shutting Down, and Restarting Microservices Container

Starting Microservices Container and Microservices Container Administrator

Microservices Container must be running in order for clients to execute services or for Microservices Container to send outbound requests. If you are using Microservices Container in a development environment, it must be running in order for your developers to build, update, and test services using the Software AG Designer.

Before starting Microservices Container, make sure there is enough free disk space on the host machine to accommodate the storage of configuration and log files on disk. Running out of disk space can affect performance and lead to errors.

For information about starting Microservices Container from the command prompt or in safe mode, see the *webMethods Integration Server Administrator's Guide*.

Starting Microservices Container on Windows

To start Microservices Container on Windows

- Click **Start > All Programs > Software AG > Start Servers > Start Integration Server > *instance_name***

Starting Microservices Container on UNIX

To start Microservices Container on UNIX

1. Log in as a non-root user.

   Note: Running the script as root might reduce the security of your system.

2. Go to the *Integration Server_directory* /profiles/*instance_name* /bin directory and run the startup.sh script file.
3. If Microservices Container has been configured to request a master password for outbound password encryption, you will be prompted for this password in a popup window or from the server console. For information about managing outbound passwords, see the *webMethods Integration Server Administrator's Guide*.

Starting Microservices Container Administrator

To start Microservices Container Administrator

1. Do one of the following:
   - Open a browser and point it to the host and port where Microservices Container is running (for example, `http://localhost:5555` or `http://EXAMPLE:4040`).

- On Windows, click **Start > All Programs > Software AG > Administration > Integration Server Administrator > *instance_name***.

2. Log on to Microservices Container Administrator with a user name and password that has administrator privileges. User names and passwords are case sensitive.

Important: The default values are Administrator/manage. For security reasons, Software AG strongly recommends you change the user name and password from the default as soon as possible.

Shutting Down Microservices Container

When you shut down Microservices Container, all active sessions also shut down. for instructions on viewing active sessions before shutting down, see *webMethods Integration Server Administrator's Guide*.

Shutting Down Microservices Container on Windows

To shut down Microservices Container on Windows

- Click **Start > All Programs > Software AG > Stop Servers > Stop Integration Server > *instance_name***

Shutting Down Microservices Container from Microservices Container Administrator on Windows or UNIX

To shut down Microservices Container from Microservices Container Administrator on Windows or UNIX

1. In Microservices Container Administrator, in the upper right corner of any screen, click **Shutdown and Restart**.
2. Specify whether you want you want Microservices Container to wait before shutting down or shut down immediately.

To...	Do this
Shut down after *number* minutes or after all client sessions are complete	Select **After all client sessions end**. Then in the **Maximum wait time** field, specify the number of minutes you want Microservices Container to wait before restarting. Microservices Container will begin monitoring user activity and then automatically shut down after all non-administrator sessions complete or after the time you specify elapses, whichever comes first.

To...	Do this
Shut down immediately	Select **Immediately**. Microservices Container and all active sessions will terminate immediately.

3. Click **Shut Down**.

Restarting Microservices Container

You should restart Microservices Container when:

- **You make certain configuration changes.** Some configuration changes require Microservices Container to be restarted before they take effect. The documentation indicates when restart is necessary.
- **You want to incorporate updated services that cannot be dynamically reloaded.** This typically occurs for non-Java services.

To restart Microservices Container

1. In Microservices Container Administrator, in the upper right corner of any screen, click **Shutdown and Restart**.
2. Specify whether you want Microservices Container to wait before restarting or restart immediately.

To...	Do this
Restart after *number* minutes or after all client sessions are complete	Select **After all client sessions end**. Then in the **Maximum wait time** field, specify the number of minutes you want Microservices Container to wait before restarting. Microservices Container will begin monitoring user activity and then automatically restart after all non-administrator sessions complete or after the time you specify elapses, whichever comes first.
Restart immediately	Select **Immediately**. Microservices Container and all active sessions will terminate immediately. Then Microservices Container restarts.

3. Click **Restart**.

3 Configuring Microservices Container

Configure Microservices Container as you would configure Integration Server. See the *webMethods Integration Server Administrator's Guide* for instructions.

You might have installed these additional components when you installed Microservices Container:

Component	For more information . . .
CentraSite Asset Publisher Support	*webMethods BPM and CAF CentraSite Metadata Help*
Common Directory Service Support	*webMethods Integration Server Administrator's Guide*
Digital Event Services Support	*webMethods Integration Server Administrator's Guide*
External RDBMS Support	*webMethods Integration Server Administrator's Guide*
Support for Tomcat	*Web Applications Developer's Guide*

You can run a microservice or a set of related microservices in a Docker container, obviating the need to purchase expensive VMs. Docker images include configuration, enabling you to deploy the exact same configuration anywhere. The Docker image can include one package or a set of related packages. For instructions on using Docker, see *webMethods Integration Server Administrator's Guide*.

Consul Support

Configuring Connections to Consul Server

Configure one or more server aliases for the public services provided in WmConsul to use to connect to a Consul server. You must create at least one server alias for the services to execute successfully. An alias name used as the *registryAlias* input parameter value for a WmConsul public service must exist in the Microservice Consul Registry Server alias list.

Note: The WmConsul package contains the public services for interacting with a Consul server.

To create an alias to the Consul server

1. Open Microservices Container Administrator.
2. in the **Packages** menu of the Navigation panel, click **Management**.
3. Click the icon for the WmConsul package.
4. On the Microservice Consul Registry Servers page, click **Create Microservice Consul Registry Server Alias**.
5. Under Microservice Consul Registry Server Alias Properties, provide the following information.

For this field...	Specify...
Alias	Name for the server alias.
Host Name or IP Address	Location of the Consul server as a host name or IP address.
Port Number	Port on which to connect to the Consul server.
Enable Consul Health Check	Whether to enable a health check for microservices registered using this alias. Consul uses the health check to keep track of the health of a registered microservice . Select **Yes** to enable health checks.
User Name	Optional. If the Consul server is configured to use a user name and password, the user name.
Password	Optional. If the Consul server is configured to use a user name and password, the password.

For this field...	Specify...
Use SSL	Whether to use a secure connection to connect to the Consul server. To use SSL, select **Yes** and then provide truststore and possibly keystore information. **Note:** If you select yes, the Consul server must be configured to accept HTTPS requests.
Keystore Alias	Optional. Keystore alias that contains the client certificates to use for the secure connection. You need to provide this information only if **Use SSL** is set to Yes and the registry server requires client certificates.
Key Alias	Optional. Key alias for the private key and certificates to use for establishing a secure connection. You need to provide this information only if **Use SSL** is set to **Yes** and the Consul server requires client certificates .
Truststore Alias	Optional. Truststore alias that contains the Consul server's certificate authority certificates. You need to provide this information only if **Use SSL** is set to Yes.

6. Click **Save Changes**.

Testing an Alias for the Consul Server

After you add an alias for a Consul server, you can test the alias to ensure that Microservices Container can establish a connection to the Consul server using the information provided in the alias.

To test a Consul server alias

1. Open Microservices Container Administrator.
2. in the **Packages** menu of the Navigation panel, click **Management**.
3. Click the icon for the WmConsul package.
4. On the Microservice Consul Registry Servers page, in the Microservce Consul Registry Server List click in the Test column for the alias you want to test.

   Microservices Container Administrator displays a status message above the list of aliases that indicates whether or not the connection is successful.

Setting the Default Alias for the Consul Server

You can identify one of the Consul server aliases as the default alias. The pub.consul.client services will use this alias to connect to the Consul server if you do not specify a different alias in the *registryAlias* input parameter.

To specify the default Consul server alias

1. Open Microservices Container Administrator.
2. in the **Packages** menu of the Navigation panel, click **Management**.
3. Click the icon for the WmConsul package.
4. On the Microservice Consul Registry Servers page, click **Change Default Alias**.
5. On the Microservice Consul Registry Servers > Change Default Alias page, in the **Default Alias** list, select the Consul server alias to use as the default.
6. Click **Update**.

Deleting a Consul Server Alias

If you no longer need an alias to a Consul server, you can delete it.

To delete a Consul server alias

1. Open Microservices Container Administrator.
2. in the **Packages** menu of the Navigation panel, click **Management**.
3. Click the icon for the WmConsul package.
4. On the Microservice Consul Registry Servers page, in the Microservice Consul Registry Server List click ✕ in the **Delete** column for the alias you want to delete. Microservices Container Administrator prompts you to confirm deleting the alias. Click **OK**.

   If you delete the default Consul server alias, Microservices Container Administrator displays a status message stating there is no longer a default Consul server alias which can prevent microservice registration.

Consul Public Services Folder

The following elements are available in this folder:

Element	Package and Description
pub.consul.client:deregisterService	WmConsul. De-registers a microservice from a Consul server. Use as a shutdown service for the package being registered as a microservice.
pub.consul.client:getAllHostsForService	WmConsul. Queries the Consul server for a list of active hosts that have registered the given microservice with Consul.
pub.consul.client:getAnyHostForService	WmConsul. Queries the Consul server for a list of active hosts that have registered the given microservice with Consul and if there are multiple hosts for this microservice, randomly return one of those hosts.
pub.consul.client:registerService	WmConsul. Registers a microservice with a Consul server. Use as a startup service for the package being registered as a microservice.

pub.consul.client:deregisterService

WmConsul. De-registers a microservice from a Consul server. Use this service as a shutdown service for the package being registered as a microservice.

Input Parameters

registryAlias	String. Optional. Alias to use to connect to a Consul server. If you do not specify this parameter, the service uses the default Consul server alias
microserviceName	String. Microservice name to de-register from the Consul server.

Output Parameters

None

Usage Notes

A package that identifies pub.consul.client:deregisterService as a shutdown service must identify WmConsul as a package dependency.

pub.consul.client:getAllHostsForService

WmConsul. Queries the Consul server for a list of active hosts that have registered the given microservice with Consul.

Input Parameters

registryAlias	String. Optional. Alias to use to connect to a Consul server. If you do not specify this parameter, the service uses the default Consul server alias
microserviceName	String. Microservice for which you want to retrieve a list of active hosts.

Output Parameters

hostList	String List. An array containing the names of hosts of the microservice. Each element is in the format *host* :*port* (for example, appserver.xyz.com:4555). If there are no hosts registered for the given microservice, returns null.

pub.consul.client:getAnyHostForService

WmConsul. Queries the Consul server for a list of active hosts that have registered the given microservice with Consul and if there are multiple hosts for this microservice, randomly return one of those hosts.

Input Parameters

registryAlias	String. Optional. Alias to use to connect to a Consul server. If you do not specify this parameter, the service uses the default Consul server alias
microserviceName	String. Microservice for which you want to retrieve an active host.

Output Parameters

hostName	String. Name of a host of the microservice, chosen randomly. The host name is in the format: *port* (for example,

appserver.xyz.com:4555). If there are no hosts registered for the given microservice, returns null.

pub.consul.client:registerService

WmConsul. Registers a microservice with a Consul server. Use this service as a startup service for the package being registered as a microservice.

Input Parameters

registryAlias	String. Optional. Alias to use to connect to a Consul server. If you do not specify this parameter, the service uses the default Consul server alias.
microserviceName	String. Microservice name to register with the Consul server.

Output Parameters

None

Usage Notes

A package that identifies pub.consul.client:registerService as a startup service must identify WmConsul as a package dependency.