Oracle® Fusion Middleware

Running Oracle WebLogic Server 12.2.1 on Docker

12*c* (12.2.1)

E66263-02

November 2015

ORACLE®

Oracle Fusion Middleware Running Oracle WebLogic Server 12.2.1 on Docker, 12*c* (12.2.1)

E66263-02

Primary Author: Jeffrey Schieli

Contributing Author: Monica Riccelli

Contributor: Bruno Borges

Contents

3 Frequently Asked Questions for Running WebLogic Server Images on Docker

Preface

This section describes the intended audience, how to use this guide, related documents, and provides information about documentation accessibility.

Audience

This document is intended for application developers who want to quickly create lightweight clustered and non-clustered WebLogic Server domain configurations on Docker, a Linux-based container technology, in order to run a single host OS or on VMs, for either development or production environments.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Documents

For more information, see the following documents in the Oracle WebLogic Server documentation set:

- *Understanding Domain Configuration for Oracle WebLogic Server*
- *Creating WebLogic Domains Using the Configuration Wizard*
- *Administering Server Startup and Shutdown for Oracle WebLogic Server*
- *Administering Node Manager for Oracle WebLogic Server*
- *Understanding the WebLogic Scripting Tool*
- *Developing Applications for Oracle WebLogic Server*
- *Deploying Applications to Oracle WebLogic Server*

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1

Getting Started

This section discusses how Oracle WebLogic Server 12.2.1 can be configured to run inside a Docker container. Docker is a Linux-based container technology that enables you to quickly create lightweight clustered and non-clustered WebLogic Server domain configurations on a single host OS or virtual machines, for either development or production environments.

Topics include:

About Docker

Docker is a platform that enables users to build, package, ship and run distributed applications. Docker users package up their applications, and any dependent libraries or files, into a Docker image.

Docker images are portable artifacts that can be distributed across Linux environments. Images that have been distributed can be used to instantiate containers where applications can run in isolation from other applications running in other containers on the same host operating system.

About WebLogic Server Images on Docker

As part of the certification, Oracle has released Dockerfiles and supporting scripts on GitHub to build images for WebLogic Server. These images are built as an extension of existing Oracle Linux images. To help you create and run WebLogic Server Docker images, the Dockerfiles and supporting scripts posted on GitHub can be used as examples to help you get started. For more information, see About the Dockerfiles and Scripts on GitHub.

To support building your Docker images, Oracle has certified using WebLogic Server 12.2.1 with various combinations of JDK versions, Oracle Linux and Red Hat Linux OS versions, Kernel versions, and Docker versions. For detailed certification information about supported WebLogic Server Docker images, see `http://www.oracle.com/technetwork/middleware/ias/oracleas-supported-virtualization-089265.html`.

Custom WebLogic Server Docker Images

You can also create your own WebLogic Server Docker images. To facilitate this process, Oracle has posted Dockerfiles and scripts on GitHub as examples that can help you to get started.

These are the prerequisites to build custom WebLogic Server Docker images:

- Supported Oracle Linux or Red Hat Linux base image
- Dockerfiles and scripts from GitHub
- Oracle WebLogic Server 12*c* (12.2.1) generic installer or Developer installer
- Corresponding supported JDK

About the Dockerfiles and Scripts on GitHub

To facilitate the building and running of WebLogic Server Docker images, Oracle has posted Dockerfiles and supporting scripts on GitHub. Download the entire directory structure to build your Oracle WebLogic Server images and start your containers. To access these files, go to `https://github.com/oracle/docker/tree/master/OracleWebLogic/`.

These Dockerfiles and scripts enable you to extend your image and create clustered and non-clustered Oracle WebLogic Server domain configurations, including both development and production running on a single host operating system or on VMs. Please note that the Dockerfiles and scripts on Github are only intended to be samples for you to write your own Dockerfiles and build your WebLogic Server images.

> **Note:** Oracle has certified WebLogic Server with the Red Hat Linux image; however, the Dockerfiles on GitHub are written to extend only the Oracle Linux image. If you want to create a WebLogic Server image and extend a Red Hat Linux image, then you must modify these Dockerfiles to do so.

Each server running in the resulting domain configurations runs in its Docker container, and can communicate as required with other servers. Other configurations and approaches are possible, as described in Building WebLogic Server Images on Docker.

What Are the Dockerfiles on GitHub?

There are two types of Dockerfiles available on GitHub for WebLogic Server 12*c* (12.2.1), located under the `/OracleWebLogic/dockerfiles/12.2.1` subdirectory:

- `Dockerfile.developer` – builds a WebLogic Server "developer" install image.
- `Dockerfile.generic` – builds a WebLogic Server "generic" domain image.

WebLogic Server Install Image Dockerfile

The Dockerfile to create an WebLogic Server install image performs the following functions:

- Extends the Oracle Linux base image.
- Installs the JDK.

- Installs WebLogic Server using either the Generic or Developer installer (depending on which Dockerfile you choose).

After creating your WebLogic Server install images, you can extend them to have a base WebLogic Server domain configured.

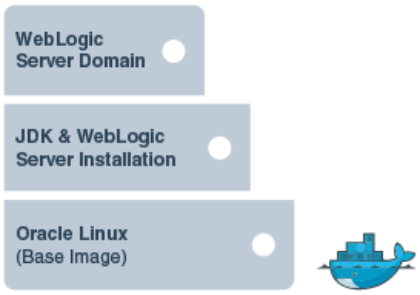
WebLogic Server Domain Image Dockerfile

The Dockerfile to create an WebLogic Server domain image performs the following functions:

- Extends the WebLogic Server install image.
- Configure a WebLogic Server domain by calling WLST scripts. The domain has one Administration Server with a JMS server and a data source, and also enables JPA 2.1 and JAX-RS 2.0.

Figure 1–1 illustrates a WebLogic Server domain image.

Figure 1–1 WebLogic Server Domain Image



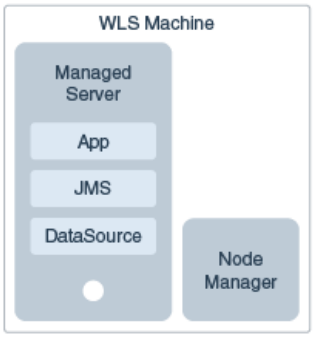
Using the Oracle WebLogic Server domain image you can create two types of containers:

- Administration server container with a single WebLogic Server Administration server, as shown in Figure 1–2.

Figure 1–2 Administration Server Container

- Managed server container with a node manager, which adds itself as a machine to the Administration Server and a Managed Server, as shown in Figure 1–3.

Figure 1–3 Managed Server with Node Manager Container



What Are the Scripts on GitHub?

The supported scripts aid in the creation of a WebLogic Server 12*c* (12.2.1) Docker image and serve as examples to extend the image with the configuration of a WebLogic Server domain. The scripts are located under the subdirectories `/OracleWebLogic/dockerfiles`, `/OracleWebLogic/samples`, and `/OracleWebLogic/samples/12c-domain/container-scripts`.

The scripts in Table 1–1 help in the creation of a WebLogic Server install image and the starting of WebLogic servers inside of a Docker container.

Table 1–1 Supported WebLogic Server Scripts for Docker on GitHub

Script	What it does
`buildDockerImage.sh`	Builds the image using the WebLogic Server installation Dockerfile instructions.
`add-machine.py`	WLST scripts to create a machine using the Managed Server container name.
`add-server.py`	WLST scripts to create a Managed Server.
`commEnv.sh`	Enables JPA 2.1 support.
`create-wls-domain.py`	WLST script configures a base domain with one Administration Server, JMS server, JSP, and data source.
`createMachine.sh`	Starts a Node Manager in the container and calls `addMachine.sh` to start Node Manager and add the Node Manager machine.
`createServer.sh`	Starts a Node Manager in the container and calls `add-server.py` to configure a Managed Server in the machine create by `add-machine.py`.
`jaxrs2-template.jar`	Template to configure JAX-RS.
`rm_containers.sh`	Removes all running containers.
`clean-up-docker.sh`	Removes all ghost containers and all ghost images.

Clustering WebLogic Server on Docker Containers

WebLogic Server uses a *machine* concept, which is an operational system with an agent, the *Node Manager*. This machine resource allows the Administration Server to create and assign Managed Servers to a domain and/or cluster, expand a domain and/or cluster, and to deploy applications and resources to the Managed Servers.

By using machines in containers, you can configure a Dynamic Cluster and easily scale up your cluster by starting new Managed Server containers. Using the WebLogic

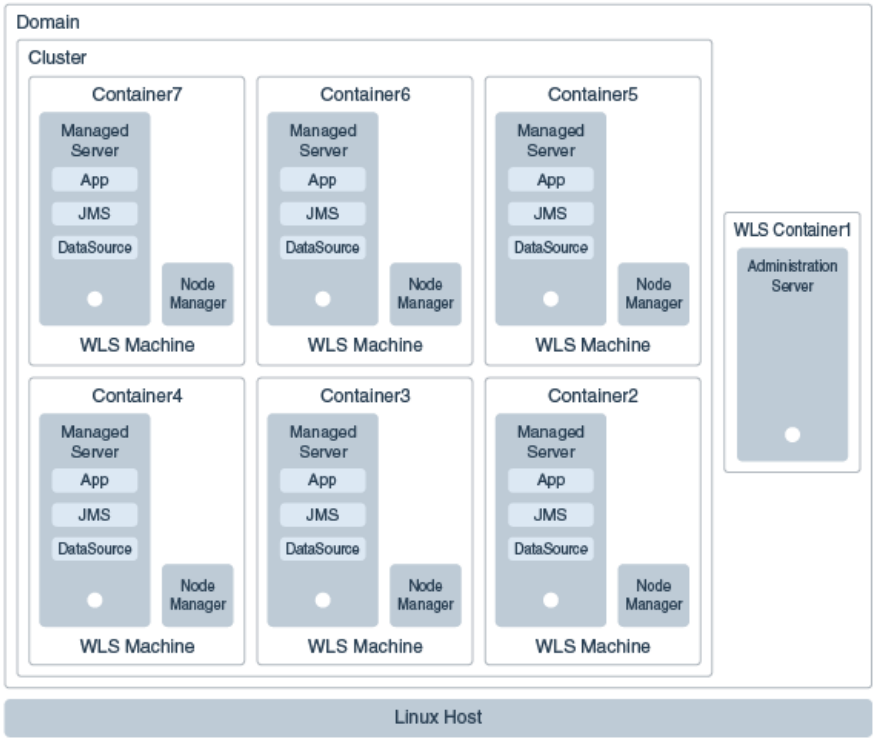
Server Scripting Tool (WLST), your cluster can quickly be scaled in and out. For more information about the Node Manager, see *Administering Node Manager for Oracle WebLogic Server*, and for more information about using WLST, refer to *Understanding the WebLogic Scripting Tool*.

The Docker containers enable you to create clustered and non-clustered WebLogic Server domain configurations across a single Linux host. Each server in the domain runs in its own Docker container and can communicate with other servers on the same host. It is important to note that Oracle does not support clustered WebLogic Servers across multiple hosts.

Clustered WebLogic Domain in Docker Containers

One topology is to configure a WebLogic Server cluster on Docker containers across a single host, as show in Figure 1–4.

Figure 1–4 WebLogic Cluster on Docker Containers Across a Single Host



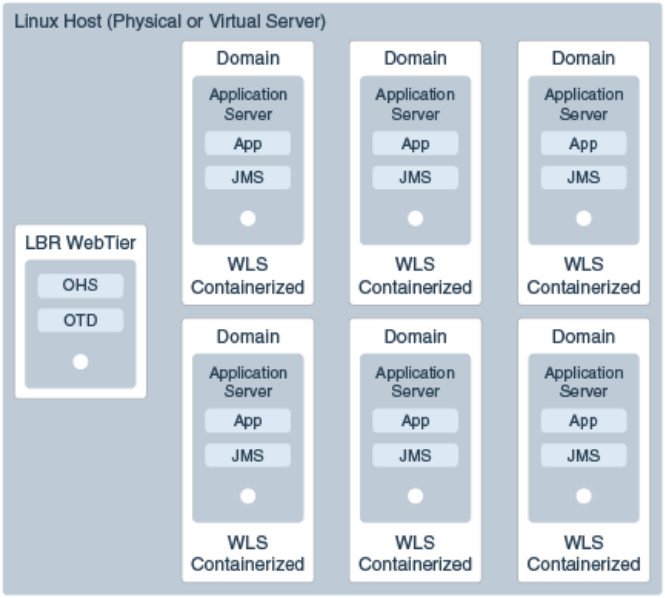
The advantages of this topology are:

- Suited for traditional WebLogic Server deployments (with a WebLogic domain and a cluster (or multiple clusters), and several servers in each cluster).
- Easy to deploy containers from WebLogic domain images.
- Easy to scale a cluster up or down.
- Convenient for developers.
- No need to install or configure anything on the host except for Docker binaries.

Non-Clustered WebLogic Server Domain in a Docker Container

A recommended topology that is in line with the "Docker way" for configuring containerized applications and services, consists of a container designed to run only an WebLogic Administration Server that contains all resources, shared libraries, and deployments. The Docker image includes all domain resources predefined, applications, and shared libraries deployed up-front, and no Managed Servers or clusters configured, as shown in Figure 1–5.

Figure 1–5 Containerized Oracle WebLogic Server Applications on a Single Host

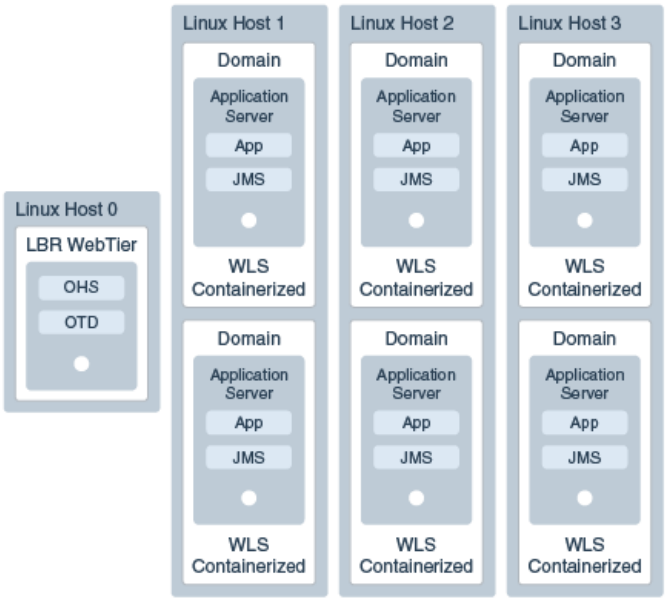


The advantages of this topology are:

- Uses the Docker "recommended" way for configuring containerized applications and services.
- Containers are easily repeatable.
- Each container is an instance of the same WebLogic domain.

The containers can all be on a single physical or virtual server Linux host as shown in Figure 1–5, or they can be on multiple physical or virtual server Linux hosts, as shown in Figure 1–6.

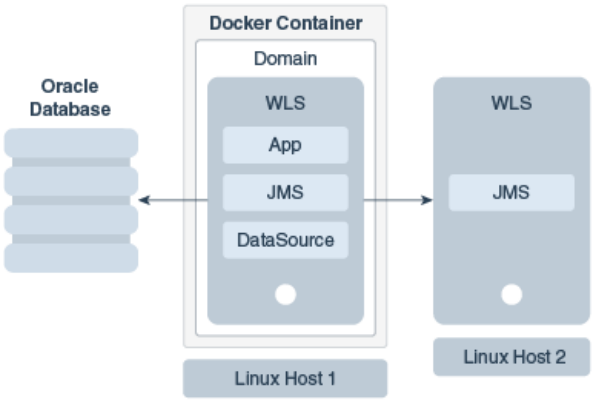
Figure 1–6 Containerized WebLogic Server Applications on a Multiple Hosts



Another possible topology is a single Oracle WebLogic Server domain running on a single container on a single Linux host communicating with a WebLogic Server instance on a remote host and a database.

Figure 1–7 illustrates a single WebLogic Server domain in a Docker containers on a single host.

Figure 1–7 WebLogic Server Domain on a Docker Container on a Single Host



2

Building WebLogic Server Images on Docker

This section discusses how to use the Dockerfiles and supporting scripts that Oracle has made available on GitHub to build your own WebLogic Server 12.2.1 Docker images.

Topics include:

Building WebLogic Server Images on Docker

Before you begin, choose the installation type you want to use, either the Generic or Developer installer, as described in About WebLogic Server Images on Docker. Then follow these steps:

1. Download the required WebLogic Server installer and JDK in the `dockerfiles/12.2.1` directory.
2. Change to the `/dockerfiles` directory and run the `buildDockerImage.sh` script as root:

   ```
   $ sudo sh buildDockerImage.sh -h
   ```

   Usage: `buildDockerImage.sh [-d]`

   where `d`: creates image based on the Developer distribution or the Generic distribution if omitted.

   > **Note:** The resulting image will *not* have a domain preconfigured. Oracle provides a separate Dockerfile and supporting scripts to extend the WebLogic Server install image and create an WebLogic Server domain image.

Samples for WebLogic Server Domain Creation

To give you an idea on how to create a domain from a custom Dockerfile to extend the WebLogic Server install image, Oracle provides some samples for WebLogic Server 12*c* (12.2.1) for both the Developer and Generic distributions. The samples are in the `samples/12c-domain` directory.

Sample Domain for WebLogic Server 12c (12.2.1)

This Dockerfile will create an image by extending `oracle/weblogic:12.2.1-dev` (from the Developer distribution). It will configure a `base_domain` with the following settings:

- JPA 2.1 enabled
- JAX-RS 2.0 deployed
- Administrator Username: `weblogic`
- Administrator Password: `welcome1`
- Oracle Linux Username: `oracle`
- Oracle Linux Password: `welcome1`
- WebLogic Server Domain Name: `base_domain`

Write Your Own Oracle WebLogic Server Domain with WLST

The best way to create your own domain, or to extend domains, is by using WLST. The WLST script used to create domains in the Dockerfile container is `create-wls-domain.py`. This script by default adds JMS resources and a few other settings.

You may want to tune this script with your own setup to create data sources and connection pools, security realms, deploy artifacts, and so on. You can also extend images and override the existing domain, or create a new one with WLST.

For more information about using WLST, refer to *Understanding the WebLogic Scripting Tool*.

Building a Sample Docker Image of a WebLogic Server Domain

To build a sample of a WebLogic Server image with a domain configured, follow these steps:

1. Make sure you have `oracle/weblogic:12.2.1-dev` image built as described in Building WebLogic Server Images on Docker. If not, change to the `/dockerfiles` directory and run the `buildDockerImage.sh` script as root:

   ```
   $ sudo sh buildDockerImage.sh -d
   ```

2. Change to the `/samples/12c-domain` directory and run the `docker build` command:

   ```
   $ sudo docker build -t samplewls:12.2.1
   ```

3. Verify that you have the image in place by running the `docker images` command:

   ```
   $ sudo docker images
   ```

Running an Administration Server Container

When you use the WebLogic Server domain image to start your container, an Administration Server will start running in the container by default. The default Administration Server name is `AdminServer`, the default port configuration is `8001`, and the default Administration Server container name is `wlsadmin`. When running

more than one domain in the same single host, you must change the Administration Server name, port number, and container name.

To start the Administration Server:

1. Execute the `docker run` command:

   ```
   $ sudo docker run -d --name=wlsadmin samplewls:12.2.1
   ```

   where `samplewls:12.2.1` is the WebLogic Server domain image tag. The samples Dockerfiles define `startWebLogic.sh` as the default CMD (command).

2. To obtain the IP address of the Administration Server Container run the `docker inspect` command:

   ```
   $ sudo docker inspect --format '{{ .NetworkSettings.IPAddress }}' wlsadmin
   ```

3. Open the Administration Server's web-based console at `http://xxx.xx.x.xx:8001/console`.

> **Note:** If you have multiple WebLogic Server domains running on the same host (such as multiple Administration Servers), change the `-name` parameter (name of the Administration Server container) and the `-p` parameter (port of the Administration Server).

Running a Managed Server Container

Managed Server containers have a Node Manager and a Managed Server running in it. These Managed Server containers communicate to an Administration Server container by linking (`--link` command) using the Administration Server container name. The Administration Server container name defaults to `wlsadmin`.

When there are more than one domain running on the same host, then the Administration Server container name needs to be unique, you need to change the Administration Server container name by using the `-name` parameter and matching the name given in the `-link` command of each Managed Server container.

There are three different ways to start a Managed Server container:

- Start Node Manager (manually):

  ```
  $ sudo docker run -d --link wlsadmin:wlsadmin <image-name> startNodeManager.sh
  ```

- Start Node Manager and create a Machine automatically:

  ```
  $ sudo docker run -d --link wlsadmin:wlsadmin <image-name>  createMachine.sh
  ```

- Start Node Manager, create a Machine, and create a Managed Server automatically:

  ```
  $ sudo docker run -d --link wlsadmin:wlsadmin <image-name>  createServer.sh
  ```

Sample Managed Server Command and Parameters

A sample `docker run` command for a Managed Server container and a listing of available parameters is as follows:

```
$ sudo docker run -d -link wlsadmin:wlsadmin \
     -p <NM Port>:5556 -p <MS Port>:<MS Port> \
     -name=<Container name> \
```

```
-e MS_HOST=<Host address where Managed Server container runs> \
-e MS_PORT=<Managed Server port> \
-e NM_HOST=<Host address where Managed Server container runs> \
-e NM_PORT=<Node Manager Port (should match the port in the -p)> \
<image name> \
<createMachine.sh, startNodeManager.sh, createServer.sh>
```

Script Variables

The supported scripts have a list of variables that must be properly configured, as defined in Table 2–1.

Table 2–1 Script Variables and Definitions

Variable	Definition
ADMIN_USERNAME	Username of the AdminServer weblogic user. Default: weblogic
ADMIN_PASSWORD	Password of ADMIN_USERNAME. Defaults to value passed during Dockerfile build. (welcome1 in the samples)
ADMIN_URL	t3 URL of the AdminServer. Default: t3://wlsadmin:8001
CONTAINER_NAME	Name of the Machine to be created. Default: node manager_ + hash of the container
NM_HOST	IP address where Node Manager can be reached. Default: IP address of the container
NM_PORT	Port of Node Manager. Default: 5556
MS_HOST	IP address where Managed Server can be reached. Default: IP address of the container
MS_PORT	Port of Managed Server. Default: 7001

Examples of Using the Script Variables

The following command will run a Managed Server container:

```
$ sudo docker run -d -link wlsadmin:wlsadmin -p 5558:5556 --name="wlsnm0" -e NM_
HOST="xx.xxx.xx.xxx" -e NM_PORT="5558" samplewls:10.3.6 createMachine.sh
```

If you want to run "Single-Host" configuration on a remote server (for example DevOps machine), you must expose ports and addresses of the Managed Servers, as shown in these examples:

```
$ sudo docker run -d --link wlsadmin:wlsadmin   -p 7003:7003
-e MS_HOST=xx.xxx.xx.xxx -e MS_PORT=7003 samplewls:12.2.1 createServer.sh

$ sudo docker run -d --link wlsadmin:wlsadmin   -p 7002:7002
-e MS_HOST= xx.xxx.xx.xxx  -e MS_PORT=7002 samplewls:12.2.1 createServer.sh
```

Note: You must assign a new, unique listen port when you create an additional Managed Server container on a host OS where a Managed Server container is already running. This prevents multiple Managed Servers running on the same host OS from listening on the same listen port.

If you used the `createServer.sh` command:

1. Access the Administration Server console at `http://admin-container-ip:8001/console`.
2. In the **Domain Structure** tree, expand **Environment**.
3. Select **Machines** to open the **Summary of Machines** page and verify that you have a Machine registered.
4. Click your registered Machine in the table, and then use the **Node Manager** and **Servers** tabs to verify that your Node Manager and Managed Server are also configured.
5. Open the **Servers** page, select the **Control** tab, and start your server.

Additional Considerations When Running WebLogic Server Images in Docker Containers

This section addresses some additional considerations when running WebLogic Server images in Docker containers.

How IP Addresses Are Handled When a Docker Container Is Restarted

When a Docker container is restarted its IP addresses change, so WebLogic Servers running in the Docker container will now have a new address. Applications, as well as others servers that were communicating with the server before the container restart, will be unable to communicate.

One solution is to configure a DNS server on Docker and configure WebLogic Server domains to use DNS names after a container restart.

How the File System Is Managed in Containers

WebLogic Server configuration files, server logs, file stores, and so on, are all kept in the container file system. When a Docker container is destroyed you will lose your entire file system. There are two alternatives that you can use to avoid losing your file system, described below and illustrated in Figure 2–1:

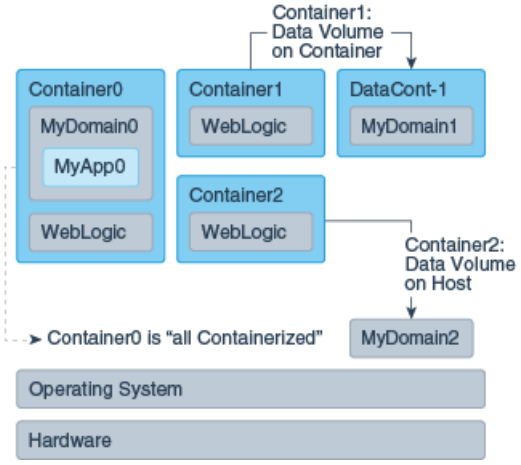
- Maintain a "data-only" container to store your domain file system. (Container1 and DataCont-1 in the figure.)
- Use the host file system to store the container's local file system (Container2 in the figure.)

Note: Container0 in the figure represents a single WebLogic Server that is stateless. Its only function is to deploy applications and resources and does not need to maintain its file system. If it is destroyed, you can just start a new one from the image.

To minimize the dependency on the file system, Oracle recommends:

- Keeping your stores, such as TLog and JMS stores, in the database.
- If you use XA Transactions, use XATransactions without TLog write, because this minimizes the writing to the TLog. See "XA Transactions without Transaction TLog Write" in *Developing JTA Applications for Oracle WebLogic Server*.

Figure 2–1 Managing the File System with Docker Containers



How Clustered WebLogic Servers Communicate in Containers

Clustered Oracle WebLogic Servers must communicate between themselves and with the Administration Server. Docker containers running on different host machines do not have the necessary visibility and access to communicate directly with other containers. Consequently, the use of WebLogic Server configurations that span multiple hosts operating systems is not supported at this time.

A possible alternative configuration is to run your entire WebLogic Server domain on a single host.

Patching and Upgrading WebLogic Server Images

To patch or upgrade your WebLogic Server images created with the WebLogic Server generic installation image, follow these steps:

1. Upgrade or patch the image by extending the WebLogic Server install Docker image.
2. Use the Docker `cp` (copy) command to copy your domain directory to a destination directory on either the host or a "data-only" container.
3. Remove the container.
4. Run the new container from the extended image (with upgrade/patch).
5. Use the Docker `cp` command to copy your domain directory back to the upgraded container.

Security Concerns Regarding Docker and Linux Containers

The following security concerns have been raised regarding Docker and Linux containers:

- One area of concern is whether it is possible to isolate code running in separate containers from each other. There are no known issues impacting the ability to run WebLogic Server in such an environment at this time.
- Another security concern is the source of the Docker images. You should only obtain Docker images from trusted sources and you need to be aware of the frequency of updates and the nature of the controls on Docker Hub.
- You should stay current with Docker and Linux technology and remain aware of security issues that are raised in each.
- Docker containers default network mode of "Bridge Networking" does not support multicast. Docker containers "Host Networking" supports multicast, but provides less isolation since it uses the host networking stack. Oracle recommends the use of unicast as the WebLogic Server clustering protocol when running in Docker containers.

3

Frequently Asked Questions for Running WebLogic Server Images on Docker

This section provides answers to frequently asked questions about running WebLogic Server 12.2.1 images in Docker containers.

Is Oracle Weblogic Server 12.2.1 certified on Oracle Linux 6.6/7 and Red Hat Linux 7 Docker images?

Yes WebLogic 12.2.1 is supported and certified on Oracle Linux 6.6/7 and Red Hat Linux 7. For detailed certification information about supported WebLogic Server Docker images, see `http://www.oracle.com/technetwork/middleware/ias/oracleas-supported-virtualization-089265.html`.

Can I create my own WebLogic Server Docker Images?

Yes, you can use the Dockerfiles and scripts posted on GitHub as examples. For more information, see "Building WebLogic Server Images on Docker" on page 2-1.

Does Oracle post WebLogic Server Docker images on the Docker Hub?

No, but Oracle has posted some Dockerfiles and supporting scripts on GitHub as samples to create WebLogic Server Docker images. For more information, see "About the Dockerfiles and Scripts on GitHub" on page 1-2.

Where is the domain file system stored for WebLogic Server images in a Docker environment?

Containers are *instances* of images. Each gets its own file system on a copy-on-write approach. Docker uses Union Filesystems. Data of a container can be persisted in three ways:

- Container Union Filesystem (default) — Copy-on-write
- Data Volume — Folder mapped to a folder on the host
- Data Volume Container — Folder mapped to a folder on another container

Is Weblogic Server on Docker supported only on a single node?

Oracle supports only single host environments for "traditional" WebLogic Server deployments (with a WebLogic domain and a cluster (or clusters), and several servers in each cluster). This is because Docker containers running on different machines (hosts) do not have the necessary visibility and access to communicate directly with other containers that are not networked together, as illustrated in Figure 1–4, "WebLogic Cluster on Docker Containers Across a Single Host".

However, for topologies that are in line with the "Docker Way" of configuring containers, Oracle supports single host *and* multiple host environments. The "Docker Way" for configuring containerized applications and services consists of a container designed to run only an Administration Server that contains all resources, shared libraries, and deployments. Such containers can all be on a single physical or virtual server Linux host (see Figure 1–5, "Containerized Oracle WebLogic Server Applications on a Single Host"), or they can be on multiple physical or virtual server Linux hosts (see Figure 1–6, "Containerized WebLogic Server Applications on a Multiple Hosts").

For more information, see "Non-Clustered WebLogic Server Domain in a Docker Container" on page 1-6.

What is the recommended procedure for patching WebLogic Server on Docker containers?

For information about patching and/or upgrading WebLogic Server on Docker, see "Patching and Upgrading WebLogic Server Images" on page 2-6.

Is there an orchestration layer to support Weblogic Server in a Docker container?

There is not a supported orchestration layer at this time.

Does Oracle support third-party software running on Docker with WebLogic Server?

Oracle only certifies WebLogic Server 12.2.1 on Docker. See `http://www.oracle.com/technetwork/middleware/ias/oracleas-supported-virtualization-089265.html`.