Dockerコンテナ上のOracle WebLogic Server

Oracle ホワイト・ペーパー | 2015 年 10 月

目次

Docker上のOracle WebLogic Server

オラクルは、Docker コンテナでの Oracle WebLogic Server（Oracle WLS）の実行をサーティファイしました。このサーティフィケーションの一環として、Oracle WebLogic Server のイメージをビルドするための Dockerfile とサポート・スクリプトを GitHub でリリースしています。これらのイメージは、既存の Oracle Linux イメージの拡張としてビルドされます。ユーザーは、これらの Oracle WebLogic Server Docker イメージを使用することも、独自のイメージを作成することもできます。

Docker は、分散アプリケーションのビルド、パッケージ化、配信、実行を行うためのプラットフォームです。Docker のユーザーは、ビルドしたアプリケーションおよびその依存ライブラリや依存ファイルを 1 つの Docker イメージにパッケージ化します。Docker イメージは、複数の Linux 環境に配布できる移植可能なアーチファクトです。配布されたイメージを使用してコンテナをインスタンス化し、そのコンテナ上でアプリケーションを実行できます。このアプリケーションは、同じホスト・オペレーティング・システム上の他のコンテナで稼働しているアプリケーションから切り離されています。

以下の表に、Oracle WebLogic Server の各バージョンでの認定状況を示します。独自の Docker イメージをビルドする際には、ここに示す Oracle WebLogic Server、JDK、Linux、Docker のバージョンの組合せを使用できます。

Oracle WebLogic Server のバージョン	JDKのバージョン	ホストOS	カーネルのバージョン	Dockerのバージョン
12.2.1.0.0	8	Oracle Linux 6 UL 6以降	Unbreakable Enterprise Kernelリリース3 （3.8.13）以降	1.7以降
12.2.1	8	Oracle Linux 7 UL 0以降	Unbreakable Enterprise Kernelリリース3 （3.8.13）以降 またはRed Hat Compatible Kernel（3.10）以降	1.7以降
12.2.1	8	Red Hat Enterprise Linux 7以降	Red Hat Enterprise Linux Kernel（3.10）以降	1.7以降
12.1.3.0.0	7/8	Oracle Linux 6 UL 5以降	Unbreakable Enterprise Kernelリリース3 （3.8.13）以降	1.3.3以降
12.1.3.0.0	7/8	Oracle Linux 7 UL 0以降	Unbreakable Enterprise Kernelリリース3 （3.8.13）以降 またはRed Hat Compatible Kernel（3.10）以降	1.3.3以降
12.1.3.0.0	7/8	Red Hat Enterprise Linux 7以降	Red Hat Enterprise Linux Kernel（3.10）以降	1.3.3以降

最新のサポート対象の Oracle WebLogic Server 構成およびサポート方針について詳しくは、Oracle Fusion Middleware のサーティフィケーション状況に関するページを参照してください。

オラクルが提供する Dockerfile とスクリプトを使用すれば、単一のホスト・オペレーティング・システムまたは複数の VM で稼働する Oracle WebLogic Server ドメインのクラスタ構成および非クラスタ構成（開発用と本番用の両方を含む）を作成できます。作成されたドメイン構成内で稼働する各サーバーは、その専用 Docker コンテナ内で実行され、必要に応じて他のサーバーと通信できます。これ以外の構成やアプローチも可能で、このホワイト・ペーパーではそれらの構成の作成方法について説明します。

Oracle WebLogic Server Dockerイメージ

オラクルでは、Oracle WebLogic Server Docker イメージをビルドするための Dockerfile とサポート・スクリプトを GitHub でリリースしています。これらのイメージは、既存の Oracle Linux 7.0 イメージの拡張としてビルドされ、JDK 7 または 8 および Oracle WebLogic Server 12c（12.2.1 および 12.1.3）のインストールを含みます。

作成可能なイメージは以下の 2 種類です。

1. 汎用インストーラで作成される Oracle WebLogic Server イメージ
2. 開発者用インストーラで作成される Oracle WebLogic Server イメージ

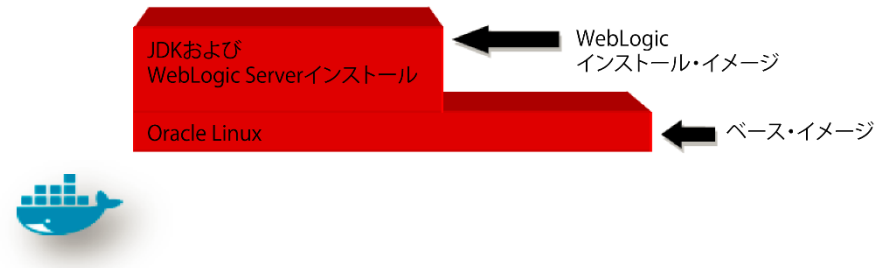

図1：Oracle WebLogic Server Dockerイメージ

カスタム・ビルドのOracle WebLogic Serverイメージ

独自の Oracle WebLogic Server Docker イメージを作成することもできます。オラクルは、作成を開始する際のサンプルとして役立つ Dockerfile とスクリプトを GitHub の Oracle WebLogic Server Dockerfile で公開しています。

カスタム WLS イメージをビルドするための前提条件は以下のとおりです。

1. Oracle Linux ベース・イメージ
2. GitHub で入手した Dockerfile とスクリプト
3. Oracle WebLogic Server の汎用インストーラまたは開発者用インストーラ
4. 対応する JDK

GitHubのDockerfileとスクリプト

Dockerfile

Oracle WebLogic Server インストール・イメージを作成するための Dockerfile は、以下の機能を実行します。

1. Oracle Linux ベース・イメージの拡張
2. JDK のインストール
3. Oracle WLS の汎用インストーラまたは開発者用インストーラを使用したサイレント・モードでの Oracle WLS のインストール

独自の Oracle WebLogic Server インストール・イメージを作成した後に、そのイメージを拡張してベースの Oracle WebLogic Server ドメインを構成できます。

Oracle WebLogic Server ドメイン・イメージを作成するための Dockerfile は、以下の機能を実行します。

- Oracle WebLogic Server インストール・イメージの拡張。
- Oracle WebLogic Server スクリプト・ツール（Oracle WLST）のスクリプトを呼び出すことによる WLS ドメインの構成。このドメインは、1 つの管理サーバー、JMS サーバー、データソースを含み、JAX-RS 2.0 を有効化します。

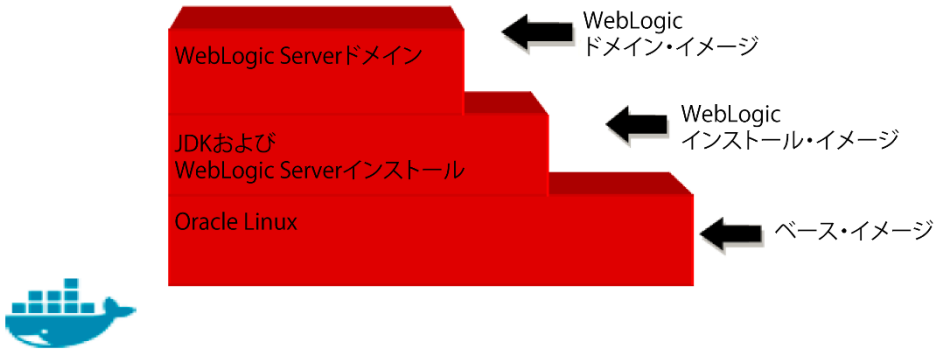

図2：Oracle WebLogic Serverドメイン・イメージ

この Oracle WebLogic Server ドメイン・イメージを使用して、以下の 2 種類のコンテナを作成できます。

1. 単一の Oracle WebLogic Server 管理サーバーを含む管理サーバー・コンテナ。
2. ノード・マネージャを含む管理対象サーバー・コンテナ。ノード・マネージャは、自身を 1 つのマシンとして管理サーバーおよび管理対象サーバーに追加します。

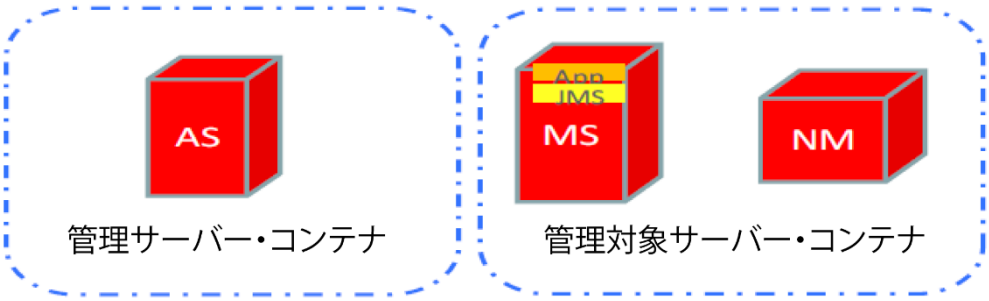

図3：Oracle WebLogic Serverコンテナの種類

スクリプト

スクリプトは、Oracle WebLogic Server イメージの作成に役立つほか、Oracle WebLogic Server ドメインの構成を使用して Oracle WebLogic Server イメージを拡張する際のサンプルとして機能します。

Dockerコンテナ上のOracle WebLogic Serverのクラスタ化

Oracle WebLogic Server にはマシンという概念があります。これは、ノード・マネージャをエージェントとする運用システムです。Oracle WebLogic 管理サーバーは、このリソースを通じて、基盤のドメインの管理対象サーバーを作成して割り当てることで、さまざまなアプリケーションやリソース向けにサーバーの環境を拡張したり、クラスタを定義したりできます。コンテナからマシンを使用することにより、新しい管理対象サーバー・コンテナを起動するだけで、動的なクラスタを簡単に作成できます。Oracle WLST を使用して、クラスタのスケールインやスケールアウトを行うことができます。

Docker コンテナを使用することで、Oracle WebLogic Server ドメインのクラスタ化構成と非クラスタ化構成を作成できます。ドメイン内で稼働する各サーバーは、その専用の Docker コンテナ内で実行され、必要に応じて同じホスト上の他のサーバーと通信できます。

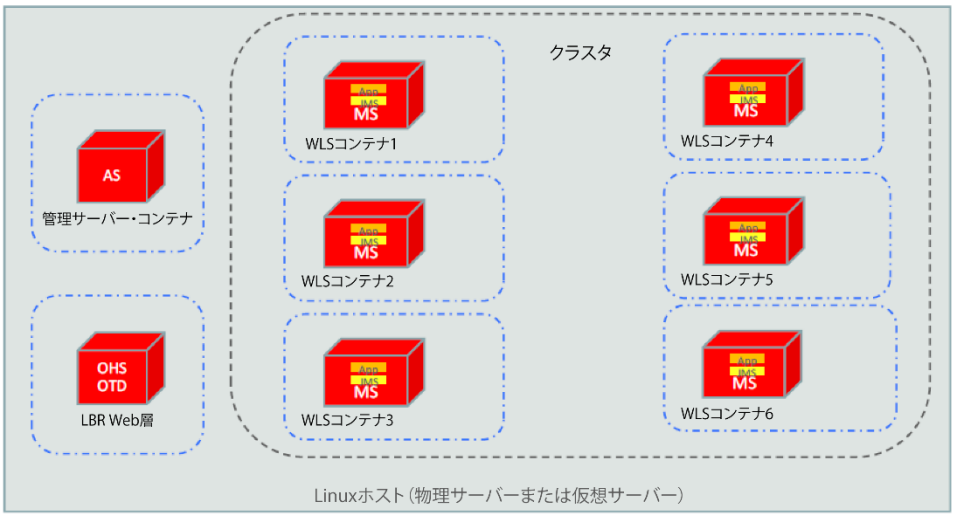

図4：単一のホスト上のDockerコンテナで稼働するOracle WebLogic Serverのクラスタ化

このトポロジには以下の利点があります。

- 従来型のデプロイメントに適しています。
- Oracle WebLogic Server ドメイン・イメージから容易にコンテナをデプロイできます。
- クラスタのスケールアップとスケールダウンが容易です。
- 開発に手間がかかりません。
- Docker バイナリ以外をホスト上でインストールまたは構成する必要がありません。

"Docker 方式"でコンテナ化されたアプリケーションとサービスに対応するトポロジは、すべてのリソース、共有ライブラリ、デプロイメントを含んだ管理サーバーのみを実行するように設計された 1 つのコンテナで構成されます。Docker イメージには、事前に定義されたすべてのドメイン・リソース、アプリケーション、あらかじめデプロイされた共有ライブラリが含まれており、管理対象サーバーまたはクラスタは構成されていません。

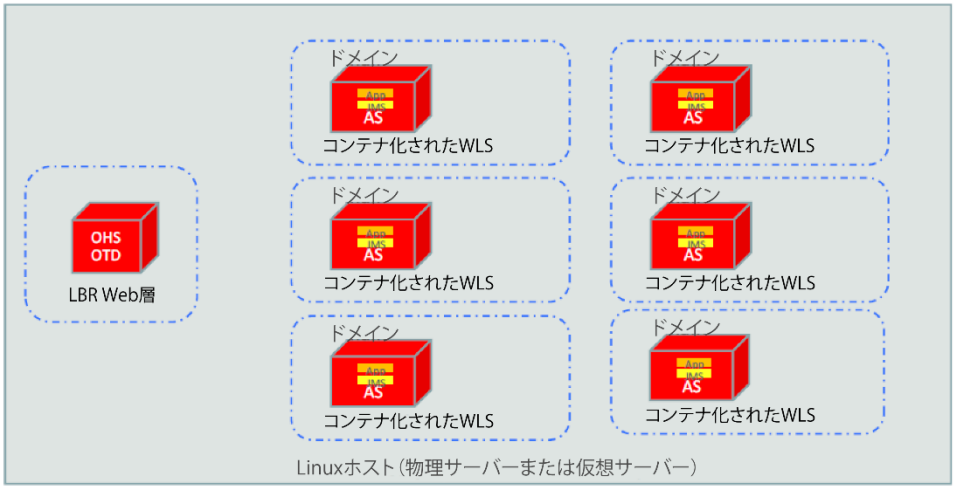

図5：単一のホスト上でコンテナ化されたOracle WebLogic Serverアプリケーション

このトポロジの利点

- アプリケーションとサービスが"Docker 方式"でコンテナ化されます。
- コンテナを簡単に再現できます。
- 各コンテナは、同じ Oracle WebLogic Server ドメインのインスタンスです。

すべてのコンテナを 1 つの物理または仮想サーバーLinux ホストに配置することも（図 5）、複数の物理または仮想サーバーLinux ホストに配置することも（図6）できます。

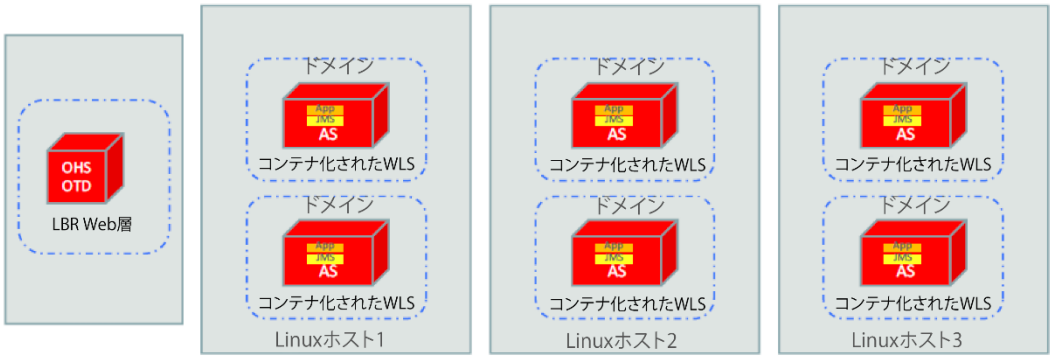

図6：複数のホスト上でコンテナ化されたOracle WebLogic Serverアプリケーション

単一の Linux ホスト上の単一のコンテナで稼働する単一の Oracle WebLogic Server ドメインが、リモート・ホスト上の Oracle WebLogic Server およびデータベースと通信するトポロジも可能です。

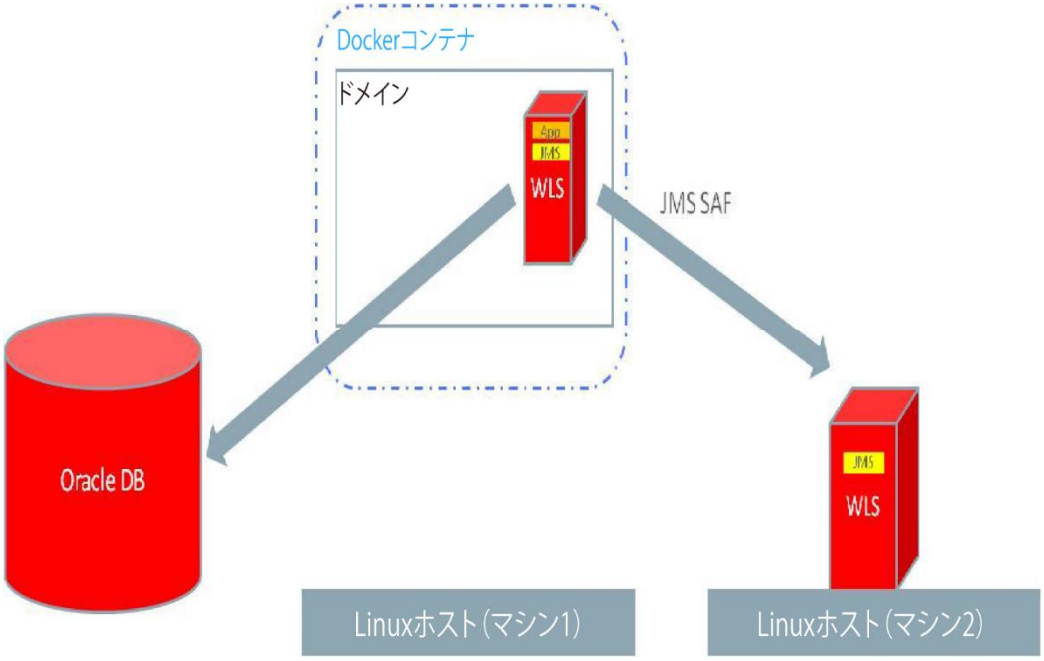

図7：単一のホスト上のDockerコンテナで稼働する単一のOracle WebLogic Serverドメイン

ビルドと実行の方法

GitHub に、Oracle WebLogic Server インストール・イメージをビルドし、このイメージを拡張して Oracle WebLogic Server ドメイン・イメージを作成するために必要な Dockerfile とサポート・スクリプトがあります。独自の Oracle WebLogic Server イメージをビルドしてコンテナを起動する場合は、このディレクトリ構造全体をダウンロードしてください。

Dockerfile

Oracle WebLogic Server 12c（12.1.3）用の 2 つの Dockerfile が

/OracleWebLogic/dockerfiles/12.1.3 サブディレクトリにあります。1 つは Oracle WebLogic Server の 'developer'（開発者向け）インストール・イメージをビルドするための Dockerfile、もう 1 つは Oracle WebLogic Server の'generic'（汎用）インストール・イメージをビルドするための Dockerfile です。

Dockerfile.developer

Dockerfile.generic

Oracle WebLogic Server 12c（12.2.1）用の同じ Dockerfile とスクリプトは、
/OracleWebLogic/dockerfiles/12.2.1 サブディレクトリにあります。

スクリプト

OracleWebLogic/samples 、 OracleWebLogic/samples/1213-domain/container-scripts 、 および OracleWebLogic/samples/1221-domain/container-scripts の下には、サポート・スクリプトがあります。コンテナ起動時に Oracle WebLogic Server インストール・イメージまたは Oracle WebLogic Server ドメイン・イメージをビルドするには、Dockerfile とともにこれらのスクリプトが必要となります。

buildDockerImage.sh –Oracle WLS インストール Dockerfile の命令を使用して WLS イメージをビルドします。

createMachine.sh –コンテナ内でノード・マネージャを起動し、addMachine.sh を呼び出してノード・マネージャを起動してノード・マネージャ・マシンを追加します。

createServer.sh –コンテナ内でノード・マネージャを起動し、add-server.py を呼び出して、add-machine.py によって作成されたマシン内で管理対象サーバーを構成します。

create-wls-domain.py –1 つの管理サーバー、JMS サーバー、JSP、データソースを含むベース・ドメインを構成する WLST スクリプトです。

add-machine.py –管理対象サーバー・コンテナ名を使用してマシンを作成する WLST スクリプトです。

add-server.py –管理対象サーバーを作成する WLST スクリプトです。rm_containers.sh –稼働中のコンテナをすべて削除します。

clean-up-docker.sh -ゴースト・コンテナとゴースト・イメージをすべて削除します。

commEnv.sh -JPA 2.1 のサポートを有効にします。

jaxrs2-template.jar-JAX-RS を構成するためのテンプレートです。

Oracle WebLogic Serverイメージのビルド

まず、インストールの種類（汎用インストーラと ZIP インストーラのどちらを使用するか）を決定し、dockerfiles/12.1.3 フォルダから必要な Oracle WebLogic Server インストーラと JDK をダウンロードします。Oracle WebLogic Server 12.2.1 の場合は、汎用インストーラとクイック・インストーラのどちらかを使用するかを決定し、dockerfiles/12.2.1 フォルダから必要な Oracle WebLogic Server インストーラと JDK をダウンロードします。dockerfiles フォルダに移動し、root として buildDockerImage.sh スクリプトを実行します。以下の説明では、12.x.x の部分を適切なバージョン（12.1.3 または 12.2.1）に読み替えてください。

$ sudo sh buildDockerImage.sh -h

使用法：buildDockerImage.sh [-d|-g] [-v] 12.x.x

パラメータ：

-d：'developer'（開発者用）ディストリビューションに基づいてイメージを作成します。

-g：'generic'（汎用）ディストリビューションに基づいてイメージを作成します。

-v：WebLogic Server のバージョンです。

注：作成されたイメージでは事前にドメインが構成されていません。Oracle WebLogic Server インストール・イメージを拡張して Oracle WebLogic Server ドメイン・イメージを作成するための Dockerfile とサポート・スクリプトが別途提供されています。

Oracle WebLogic Serverドメインの作成例

カスタム Dockerfile からドメインを作成して Oracle WebLogic Server インストール・イメージを拡張する方法を示すために、Oracle WebLogic Server 12c 用の開発者用ディストリビューションと汎用ディストリビューションのサンプルをいくつか提供しています（samples/12xx- domain フォルダ内）。

Oracle WebLogic Server 12c用のサンプル・ドメイン

この Dockerfile は、oracle/weblogic:12.x.x-dev（開発者用ディストリビューション内）を拡張してイメージを作成します。以下の設定で base_domain が構成されます。

- JPA 2.1 を有効化
- JAX-RS 2.0 をデプロイ
- 管理者のユーザー名：weblogic
- 管理者のパスワード：welcome1
- Oracle Linux のユーザー名：oracle
- Oracle Linux のパスワード：welcome1
- Oracle WebLogic Server ドメイン名：base_domain

Oracle WLSTを使用した独自のOracle WebLogic Serverドメインの作成

独自のドメインを作成または拡張する最適な方法は、Oracle WLST を使用することです。Dockerfile でドメインを作成するために使用する WLST スクリプトは、create-wls-domain.py です。このスクリプトでは、デフォルトで JMS リソースと他のいくつかの設定が追加されます。このスクリプトを独自の設定で調整して、データソース、接続プール、セキュリティ・レルムを作成したり、アーチファクトをデプロイしたりできます。また Oracle WLST では、イメージの拡張、既存のドメインのオーバーライド、ドメインの新規作成などを行うこともできます。

Oracle WebLogic ServerドメインのサンプルDockerイメージのビルド

ドメインが構成された Oracle WebLogic Server イメージのサンプルを試すには、以下の手順を実行します。

oracle/weblogic:12.x.x-dev イメージがビルド済みであることを確認します。ビルドされていない場合は、dockerfiles に移動して以下のコマンドを実行します。

$sudo sh buildDockerImage.sh [-d|-g]

samples/12c-domain フォルダに移動して以下のコマンドを実行します。

$sudo docker build -t samplewls:12.x.x

以下のコマンドを実行して、このイメージが作成されたことを確認します。

$sudo docker images

Oracle WebLogic Server管理サーバー・コンテナの実行

Oracle WebLogic Server ドメイン・イメージを使用してコンテナを起動すると、デフォルトではそのコンテナ内で管理サーバーの実行が開始されます。デフォルトの管理サーバー名は "AdminServer"、デフォルトのポート構成は 8001、デフォルトの管理サーバー・コンテナ名は "wlsadmin"です。同じ単一のホストで複数のドメインが稼働している場合は、管理サーバー名、ポート、およびコンテナ名を変更する必要があります。

Oracle WebLogic 管理サーバーを起動するには、**docker run -d samplewls:12.x.x** コマンドを呼び出します（"samplewls:12.x.x"は Oracle WebLogic Server ドメイン・イメージのタグ）。サンプルの Dockerfile では、startWebLogic.sh がデフォルトの CMD（コマンド）として定義されています。

$ sudo docker run -d --name=wlsadmin samplewls:12.x.x

管理サーバー・コンテナの IP アドレスを取得するには、以下のコマンドを実行します。

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

戻り値の例：**xxx.xx.x.xx**

これで、管理サーバーWeb コンソール（http://xxx.xx.x.xx:8001/console）にアクセスできるようになります。

同じホストで複数の Oracle WebLogic Server ドメインが稼働している場合（管理サーバーが複数ある場合）は、–name（管理サーバー・コンテナの名前）と–p（管理サーバーのポート）を変更してください。

Oracle WebLogic Server管理対象サーバー・コンテナの実行

管理対象サーバー・コンテナには、ノード・マネージャとノード・マネージャ内で稼働している管理対象サーバーが含まれています。管理対象サーバー・コンテナは、管理サーバー・コンテナ名を使用してリンクする（--link コマンド）ことで、管理サーバー・コンテナと通信します。デフォルトの管理サーバー・コンテナ名は、"wlsadmin"です。同じホストで複数のドメインが稼働しているために管理サーバー・コンテナ名を一意にする必要がある場合は、–name パラメータを使用し、各管理対象サーバー・コンテナの--link コマンドで指定する名前と一致させることで、管理サーバー・コンテナ名を変更する必要があります。

管理対象サーバー・コンテナを起動するには、3 つの方法があります。ノード・マネージャを手動で起動します。

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

ノード・マネージャを起動し、自動的にマシンを作成します。

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

ノード・マネージャを起動し、自動的にマシンを作成し、管理対象サーバーを作成します。

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

使用できるパラメータは以下のとおりです。

```
$ 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>
```

スクリプトには一連の変数が含まれており、それぞれを適切に構成する必要があります。

変数	意味
ADMIN_USERNAME	管理サーバーの'weblogic'ユーザーのユーザー名。デフォルト：weblogic
ADMIN_PASSWORD	ADMIN_USERNAMEのパスワード。デフォルト：Dockerfileビルド時に渡された値。（サンプルでは'welcome1'）
ADMIN_URL	管理サーバーのt3 URL。デフォルト：t3://wlsadmin:8001
CONTAINER_NAME	作成するマシンの名前。デフォルト：node manager_ + コンテナのハッシュ
NM_HOST	ノード・マネージャにアクセスするためのIPアドレス。デフォルト：コンテナのIPアドレス
NM_PORT	ノード・マネージャのポート。デフォルト：5556
MS_HOST	管理対象サーバーにアクセスするためのIPアドレス。デフォルト：コンテナのIPアドレス
MS_PORT	管理対象サーバーのポート。デフォルト：7001

例：

リモート・サーバーで"単一ホスト"構成を実行する場合は、管理サーバー、管理対象サーバー、およびノード・マネージャのポートとアドレスを公開する必要があります。

以下のコマンドを実行すると、管理対象サーバー・コンテナが実行されます。

$ sudo docker run -d –link wlsadmin:wlsadmin -p 5556:5556 --name="wlsnm0" -e NM_HOST="xx.xxx.xx.xxx" -e NM_PORT="5556" samplewls:12.x.x createMachine.sh

$ sudo docker run -d --link wlsadmin:wlsadmin -p 7003:7003

-e MS_HOST=xx.xxx.xx.xxx -e MS_PORT=7003 samplewls:12.x.x 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.x.x createServer.sh

管理対象サーバー・コンテナがすでに稼働しているホスト OS で追加の管理対象サーバー・コンテナを作成する場合は、新しい一意のリスニング・ポートを割り当てる必要があります。これにより、同じホスト OS で稼働する複数の管理対象サーバーが、同じリスニング・ポートでリスニングすることを防ぎます。

createServer.sh コマンドを使用した場合

1. 管理サーバー・コンソール（http://<admin-container-ip>:8001 /console）にアクセスします。
2. Environment→Machines に移動すると、マシンが登録されています。
3. ノード・マネージャと管理対象サーバーも構成されています。
4. Environment→Machines→Servers に移動し、「Control」タブをクリックしてサーバーを起動します。

Oracle WebLogic Server Dockerコンテナとリモート・ホスト上のサーバーの通信

管理サーバー・コンテナを 1 つ実行し、そのコンテナがリモート・ホストで稼働する Oracle WebLogic Server と通信するというトポロジも可能です。--add-host を使用して、ローカル・コンテナの IP アドレスではなくコンテナが稼働しているホストの IP アドレスをコンテナに指定するようにします。

```
$ sudo docker run -d -p 8001:8001 --net=host \
    --add-host=hostname:<host ip address where container is running> \
    --name wlsadmin samplewls:12.x.x
```

この構成が機能するためには、以下の構成が必要です。

- Docker コンテナ内の管理サーバーのリスニング・アドレスを構成すること。
- リモート・ホスト内の管理サーバーのリスニング・アドレスを構成すること。
- クライアントがホストの IP アドレスを使用して JNDI ルックアップ用の初期コンテキストを取得すること。

DockerでOracle WebLogic Serverを実行する場合のその他の考慮事項

- Docker コンテナが再起動されてその IP アドレスが変更されると、その Docker コンテナ内で稼働する Oracle WebLogic Server のアドレスが新しくなります。コンテナの再起動の前にそのサーバーと通信していたアプリケーションやその他のサーバーは通信できなくなります。コンテナ再起動後の IP アドレスの変更に対処するためには、Docker で DNS サーバーを構成し、DNS 名を使用するように WLS ドメインを構成します。
- Oracle WebLogic Server 構成、サーバー・ログ、ファイル・ストアなどは、すべてコンテナ・ファイル・システムに保管されます。Docker コンテナが破損すると、ファイル・システム全体が失われます。これに対処するための選択肢が 2 つあります。
    - ホストのファイル・システムを使用してコンテナのローカル・ファイル・システムを保管する。
    - ドメインのファイル・システムを保管するための"データ専用"コンテナを保持する。

  ファイル・システムへの依存を最小限に抑えるために、以下の対策を推奨します。
    - TLog ストアや JMS ストアなどのストアをデータベースに保管する。
    - XA トランザクションを実行する場合は、"TLog に書き込まない XA トランザクション"を使用して、TLog への書込みを最小限に抑える。

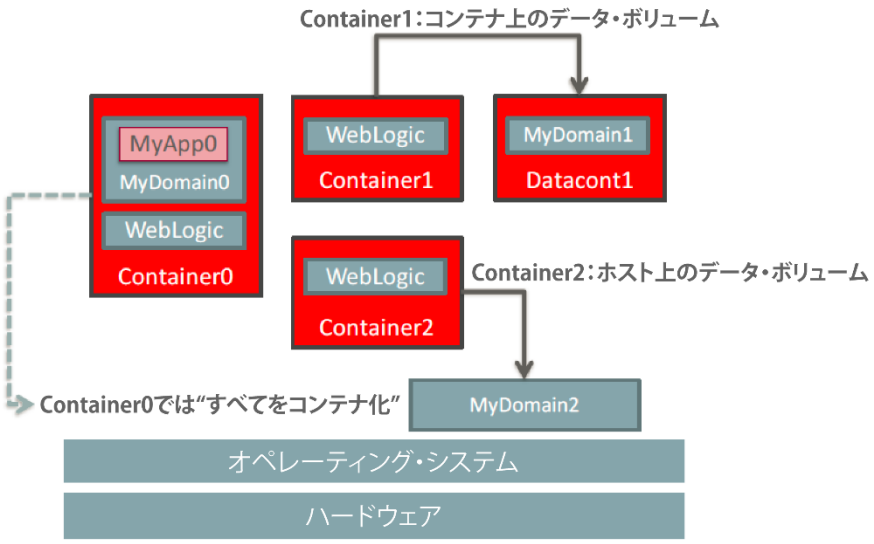

図10：コンテナまたはホスト内のDockerデータ・ボリュームで稼働するOracle WebLogic Server

- クラスタ化された Oracle WebLogic Server は、互いに通信し、管理サーバーとも通信する必要があります。別々のホスト・マシンで稼働する Docker コンテナには、他のコンテナと直接通信するために必要な可視性やアクセス権がありません。このため、現時点では、複数のホスト・オペレーティング・システムにまたがる Oracle WebLogic Server 構成を使用することができません。これに代わる構成として、Oracle WebLogic Server ドメイン全体を単一のホストで実行することが考えられます。
- Oracle WebLogic Server 汎用インストール・イメージで作成された Oracle WebLogic Server のパッチ適用またはアップグレードを行うには、以下の手順を実行します。
  1. Oracle WebLogic Server インストール Docker イメージを拡張して、アップグレードまたはパッチ適用を行います。
  2. Docker cp（コピー）コマンドを使用して、ホストまたは"データ専用"コンテナにドメイン・フォルダをコピーします。
  3. コンテナを削除します。
  4. アップグレードまたはパッチ適用によって拡張したイメージから、新しいコンテナを実行します。
  5. Docker cp（コピー）コマンドを使用して、アップグレードしたコンテナにドメイン・フォルダをコピーして戻します。
- Docker コンテナと Linux コンテナには、セキュリティに関する以下の懸念があります。
  - 別々のコンテナで実行されているコードを互いに分離できるかどうかが懸念されます。現時点では、このような環境での Oracle WebLogic Server の実行に影響する既知の問題はありません。
  - Docker イメージのソースに関するセキュリティ上の懸念もあります。Docker イメージは信頼できるソースから取得する必要があり、更新の頻度や Docker Hub での管理の特性を理解しておく必要があります。

- Docker と Linux のテクノロジーに関する最新情報を常に把握し、それぞれで発生しているセキュリティの問題を認識しておく必要があります。
- Docker コンテナのデフォルトのネットワーク・モードである"ブリッジ・ネットワーク"では、マルチキャストがサポートされません。Docker コンテナの"ホスト・ネットワーク"ではマルチキャストがサポートされますが、ホストのネットワーク・スタックが使用されるために独立性は低くなります。Docker コンテナで稼働する Oracle WebLogic Server のクラスタ化プロトコルには、ユニキャストを使用することを推奨します。

結論

Docker のアーチファクトは複数の Linux 環境への移植が可能で、配布も容易であるため、Docker テクノロジーは運用の簡素化とコストの削減をもたらすことが期待されます。

オラクルは、Docker コンテナでの Oracle WebLogic Server の実行を認定し、Docker コンテナで稼働する Oracle WebLogic Server 構成の作成をサポートするイメージ、Dockerfile、スクリプトを提供することで、お客様の高まる関心に応えてきました。これらの機能が役立つことを願い、今後も Docker 環境のサポート範囲を強化していきます。

Oracle Corporation, World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065, USA

海外からのお問い合わせ窓口
電話：+1.650.506.7000
ファクシミリ：+1.650.506.7200

CONNECT WITH US

blogs.oracle.com/oracle
facebook.com/oracle
twitter.com/oracle
oracle.com

Hardware and Software, Engineered to Work Together

Docker コンテナ上の Oracle WebLogic Server
2015 年 10 月
WebLogic Server PM、Monica Riccelli
Technical Advisor、Bruno Borges

Oracle is committed to developing practices and products that help protect the environment