Red Hat OpenShift 上での

NetCOBOL 運用環境の構築手順

2017/03/10

富士通株式会社

1. はじめに

本資料は、NetCOBOL運用環境をRed Hat OpenShift V3上に構築する手順について説明します。

1.1 対象製品

本資料の対象製品は以下です。

- Linux (64bit)版 NetCOBOL Base Edition 運用パッケージ (64bit) V11.1.0

1.2 必要環境・媒体

以下の環境、媒体が必要です。

- Red Hat Enterprise Linux 7
- Red Hat OpenShift Container Platform（Red Hat OpenShift Onlineは対象外）
- Linux (64bit) 版 NetCOBOL Base Edition 運用パッケージ(64bit)のCD-ROM

また、Dockerイメージを作成する環境のディスクに約2GBの空き容量が必要です。
NetCOBOL運用環境のDockerイメージのサイズは約370MBです。

1.3 検証環境

本資料の手順は以下を使用して検証しています。

- Red Hat Enterprise Linux 7.3
- Red Hat OpenShift Container Platform 3.3
- Linux (64bit)版NetCOBOL Base Edition 開発・運用パッケージ(64bit) V11.1.0

1.4. 前提知識

本資料を読む場合、以下の知識が必要です。

- Red Hat Enterprise Linuxに関する基本的な知識
- Dockerに関する基本的な知識
- Red Hat OpenShiftに関する基本的な知識
- NetCOBOLに関する基本的な知識

1.5. 商標

Linux(R)は、Linus Torvalds 氏の日本およびその他の国における登録商標または商標です。
Red Hat(R)、OpenShift(R)は、米国およびその他の国において登録されたRed Hat, Inc. の商標です。その他、本資料に記載されている会社名および製品名は、それぞれ各社の商標または登録商標です。

2. 概要

Red Hat OpenShift 上での、NetCOBOL運用環境の構築は以下の手順で構成されます。
1. NetCOBOL運用環境のDockerイメージ作成
2. Red Hat OpenShift における作成したDockerイメージの実行

3. NetCOBOL 運用パッケージ Docker イメージ作成

本手順はrootユーザで実施してください。
本手順は以下の手順で構成されます。
1. NetCOBOL運用パッケージのDockerイメージの作成
2. 1のDockerイメージにCOBOLアプリケーションを配備したDockerイメージ作成

3.1. NetCOBOL 運用パッケージの Docker イメージの作成

3.1.1. NetCOBOL 運用パッケージ CD-ROM のマウント

NetCOBOL運用パッケージCD-ROMをマウントします。以下は、運用パッケージCD-ROMを/mnt にマウントする例です。

```
# mount -t iso9660 -r /dev/cdrom /mnt
```

3.1.2. NetCOBOL 運用パッケージをインストールしたコンテナの作成

Red Hat Enterprise Linux 7.3のDockerイメージから、netcobolbasecontainerという名前のDockerコンテナを作り、乗り込みます。この際、3.1.1で運用パッケージCD-ROMをマウントしたディレクトリを、Dockerコンテナ内の/home/netcobolCDROMにマウントします。

```
# docker run -it --name="netcobolbasecontainer" -v /mnt:/home/netcobolCDROM
registry.access.redhat.com/rhel7.3 /bin/bash
```

以下のコマンドを実行し、Dockerコンテナ内でNetCOBOL運用パッケージをインストールします。

```
# /home/netcobolCDROM/INSTALL.sh -s
```

"Successfully installed"の出力を確認したら、以下のコマンドでシェルを終了し、Docker

コンテナを停止します。

```
# exit
```

3.1.3. Docker イメージの作成

3.1.2 で作成した Docker コンテナから Docker イメージを作成します。以下は netcobolbasecontainer という名前の Docker コンテナから"netcobolbaseimage"という名前の Docker イメージを作成する例です。

```
# docker commit netcobolbasecontainer netcobolbaseimage
```

3.1.4. NetCOBOL 運用パッケージ CD-ROM のアンマウント

NetCOBOL運用パッケージCD-ROMをアンマウントします。以下は、/mnt にマウントした運用パッケージCD-ROMをアンマウントする例です。

```
# umount /mnt
```

3.2. COBOL アプリケーションを配備した Docker イメージ作成

3.2.1. Docker コンテナに配備するアプリケーションの準備

Dockerコンテナ起動時に実行するシェルスクリプト"start.sh"と、シェルスクリプト内で起動するCOBOLアプリケーションを準備します。
図1のシェルスクリプトを作成します。以下はCOBOLランタイムの初期設定を行うシェルと、COBOLアプリケーション"hellocobol"を実行する例です。

図1: Dockerコンテナ起動時に実行するシェルスクリプト"start.sh"の例

```
#!/bin/bash
. /opt/FJSVcbl64/config/cobolrt.sh
/home/netcobol/hellocobol
```

COBOLアプリケーションを準備します。今回の手順書では、標準出力に"Hello!　COBOL"と出力するアプリケーション"hellocobol"を作成します。
図2のCOBOLソースコード"HelloCobol.cob"を作成します。

図2: hellocobolのソースコード"HelloCobol.cob"

```
000010* Copyright 2017 FUJITSU LIMITED
000020 IDENTIFICATION DIVISION.
000030 PROGRAM-ID. HELLOCOBOL.
000040 DATA DIVISION.
```

```
000050 WORKING-STORAGE SECTION.
000060 PROCEDURE DIVISION.
000070         DISPLAY "Hello! COBOL".
000080 END PROGRAM HELLOCOBOL.
```

以下のコマンドを実行し、COBOLソースコード"HelloCobol.cob"を翻訳・リンクして COBOLアプリケーション"hellocobol "を作成します。

```
cobol -dy -M -o    hellocobol HelloCobol.cob
```

3.2.2. Dockerfile の作成

図3 の内容のDockerfileを作成します。以下は、3.1で作成した"netcobolbaseimage"をベースイメージに使用した例です。

図3: Dockerfileの例

```
FROM netcobolbaseimage

#環境設定
ENV LANG ja_JP.UTF-8
ENV LANGUAGE ja_JP:ja
RUN /bin/localedef -f UTF-8 -i ja_JP ja_JP.UTF-8

WORKDIR /home/netcobol

# COBOLアプリケーションのコピー
COPY hellocobol hellocobol
RUN chmod ugo+rx hellocobol

#コンテナ起動時に実行するシェルスクリプトのコピー
COPY start.sh start.sh
RUN chmod ugo+rx start.sh

#ユーザの指定
USER 1000

CMD ["/home/netcobol/start.sh"]
```

3.2.3. Docker イメージの作成

以下の手順を実施して、NetCOBOL運用環境のDockerイメージを作成します。

1. 以下を同じディレクトリに配置します。

- 3.2.1で準備したシェルスクリプト"start.sh"
- 3.2.1で準備したCOBOLアプリケーション"hellocobol"
- 3.2.2で作成した"Dockerfile"

2. Dockerイメージを作成します。
以下は、Dockerfileをディレクトリ"/docker"に配置して、名前"netcobolimage "の
Dockerイメージを作成する例です。

```
# docker build -t netcobolimage .
```

3. DockerイメージからDockerコンテナを起動します。
以下は、Dockerイメージ"netcobolimage"からDockerコンテナ"netcobolcontainer"
を起動する例です。

```
# docker run -it --name netcobolcontainer netcobolimage
```

Dockerコンテナの起動に成功すると、以下の文字がコンソールに出力されます。

```
Hello! COBOL
```

4. Red Hat OpenShift 上での NetCOBOL アプリケーションの実行

以下の手順を実施して、COBOLアプリケーション をRed Hat OpenShiftで実行します。
本手順は、Dockerイメージを作成した環境で実行します。
1. シェルスクリプト"start.sh"を修正します。
2. Dockerイメージを再作成します。
3. Red Hat OpenShift からアクセス可能なリポジトリに、Dockerイメージを登録します。

4.1.シェルスクリプトの修正

シェルスクリプト"start.sh"を図4のように修正し、OpenshiftにDockerコンテナを配備した際にDockerコンテナが起動し続けるようにします。これは、OpenshiftにDockerコンテナを配備する際に、OpenshiftがDockerコンテナの起動と停止を繰り返してしまう状態（CrashLoopBackOff）になることを防ぐためです。

図4:修正したシェルスクリプトの例

```
#!/bin/bash
. /opt/FJSVcbl64/config/cobolrt.sh

/home/netcobol/hellocobol

while :
do
        sleep 1
done
```

4.2. Docker イメージの再作成

4.1 のシェルスクリプトの修正を Docker イメージに反映するために、Docker イメージを再作成します。

```
# docker build -t netcobolimage .
```

4.3. Red Hat OpenShift からアクセス可能なリポジトリに Docker イメージを登録

1. Openshiftからアクセス可能なリポジトリに、Dockerイメージを登録します。
以下は、Dockerイメージを"192.168.100.102:5000/netcobolimage:latest"として登録する例です。
リポジトリのIPアドレスは、ご自身の環境でアクセス可能なリポジトリのIPアドレスに変更してください。

```
# docker tag netcobolimage:latest 192.168.100.102:5000/netcobolimage:latest
# docker push 192.168.100.102:5000/netcobolimage:latest
```

2. Red Hat OpenShift にログインします。

3. Red Hat OpenShift 上で、リポジトリに登録したDockerイメージを実行します。
以下は、Dockerイメージ"192.168.100.102:5000/netcobolimage:latest"を"netcobolcontainer"として実行する例です。
リポジトリのIPアドレスは、ご自身の環境でアクセス可能なリポジトリのIPアドレスに変更してください。

```
# oc new-app --docker-image=192.168.100.102:5000/netcobolimage:latest --name
netcobolcontainer
```

4. Dockerコンテナの起動時にcobolアプリケーションが実行されたことを確認するために、

podの出力ログを確認します。ログを確認するのは、アプリケーション配備時にコンテナの標準出力がコンソールに出力されないためです。以下は、pod名が”netcobolcontainer-1-d2tp1”の場合の例です。

```
# oc logs netcobolcontainer-1-d2tp1
```

今回の例では、以下のようにログが出力され、COBOLアプリケーションが動作したことが確認できます。

```
Hello! COBOL
```

以上