導入

DockerEngine導入ガイド（Mac OS X 用）

この文章は Mac OS X の利用者向けです。Mac OS X を使っていなければ Linux や Windows 向けの文章をご覧ください。

導入ガイドの読者対象は、専門家でなくとも Docker を学ぶことに興味のある方です。導入ガイドでは、簡単な作業を通して Docker 機能の基本を学びます。導入ガイドで学ぶ内容は、次の通りです。

- Docker ツールボックスを使い、 Docker ソフトウェアをインストール。
- Docker エンジンを使い、コンテナ内でイメージを実行。
- Docker Hub（ハブ） 上でイメージを探す。
- イメージを取得し、コンテナとして実行。
- Docker Hub アカウントとイメージ・リポジトリを作成。
- イメージの作成。
- 誰でも使えるように Docker Hub にイメージを送信。

進める上で発生する問題が減るように、この導入ガイドは利用者テストを経ています。個人で色々試すよりも、導入ガイドを進める方が、最も成功に至るチャンスです。なお、ガイドを終えるまでにかかる時間は約 45 分です。

ご確認ください

この導入ガイドでは、Docker　エンジンのコマンドライン・ツール（CLI）をターミナル・ウインドウ上で使います。コマンドラインの利用経験がなくても構いません。ですが、コマンドラインの開き方や、コマンドの入力方法には慣れておいた方が良いでしょう。

目次

1.1 DockerToolbox をインストール

Mac OS X ユーザは Docker ツールボックス（Toolbox）を使い、 Docker のソフトウェアをインストールできます。Docker ツールボックスには、以下の Docker ツール群が入っています。

- Docker CLI（シーエルアイ） クライアントは Docker Engine（エンジン）でコンテナやイメージを作成。
- Docker Machine（マシン）は Windows 端末から Docker Engine に命令。
- Docker Compose（コンポーズ）は `docker-compose` コマンドを実行。
- Kitematic（カイトマティック）は Docker の GUI（グーイ） 。
- Docker QuickStart（クイックスタート）は Docker コマンドライン環境を設定済みのシェル。
- Oracle（オラクル） VM VirtualBox（バーチャルボックス）

Docker Engine デーモンは Linux 特有の kernel（カーネル） 機能を使います。そのため、OS X では Docker Engine を直接操作できません。その代わりに `docker-machine` コマンドで自分のマシン上に小さな Linux 仮想マシンを作成し、そこに接続します。この仮想マシンのホストを使い、 Mac システム上で Docker Engine を動かします。

ステップ１：バージョンの確認

Mac で Docker ソフトウェアを実行するには OS X 10.8「Mountain Lion」以上のバージョンが必要です。OS のバージョンは次の手順で確認します。

1. アップルのメニュー（画面左上の林檎マーク）から［ この Mac について ］をクリックします。

画面上にある OS X の後ろ側にある文字がバージョン番号です。

2. 適切なバージョンであれば、次のステップに進みます。

サポートしているバージョンでなければ、オペレーティング・システムのアップグレードをご検討ください。

ステップ２：Docker Toolbox のインストール

1. Docker Toolbox[*1] のページに移動します。

2. インストーラのダウンロード先をクリックします。

3. Docker Toolbox をインストールします。パッケージをダブル・クリックするか、右クリックして開いたポップアップ・メニューから［ 開く ］を選びます。

インストーラが起動すると、何をインストールするかの概要説明が表示されます。

4. ［ Contine ］を押すと Toolbox のインストールに進みます。

インストーラ画面には標準インストール対象のカスタマイズが表示されます。

*1 https://www.docker.com/toolbox

デフォルトは、標準的な Docker Toolbox のインストールです。

- Docker ツール群のバイナリを `/usr/local/bin` にインストールします。
- これらのバイナリを、全てのユーザで利用可能にします。
- 既存の Virtual Box がインストールされていれば、更新します。

現時点では、これらのデフォルトを変更しません。

5. ［ Install ］を押して、標準インストールを進めます。

画面上にはパスワード入力を促す画面が表示されます。

6. インストールを続けるためにパスワードを入力します。

インストールが終わると、ショートカットの作成を訊ねてきます。ここでは、このまま［ Continue ］(続ける) を押します。

そして［ Close ］を押してインストーラを終了します。

ステップ３：インストールの確認

Docker コンテナを実行するには：

- 新しい Docker Engine のホストを作成します（あるいは、既存のホストを起動します)。
- 環境変数を新しい仮想マシンに切り替えます。
- docker クライアントを使い、コンテナの作成、読み込み、管理を行します。

マシンを作成後は、必要な時に何度でも再利用できます。これは Virtual Box の仮想マシンと同様で、共通の設定を使います。

1. Launchpad を起動し、Docker Quickstart Terminal（クイックスタート・ターミナル）のアイコンを探します。

2. アイコンをクリックし、Docker Quickstart ターミナル・ウインドウを起動します。

ターミナルでは、Docker ツールボックスがセットアップに関する様々な情報を表示します。

```
Last login: Sat Jul 11 20:09:45 on ttys002
bash '/Applications/Docker Quickstart Terminal.app/Contents/Resources/Scripts/start.sh'
Get
http:///var/run/docker.sock/v1.19/images/json?all=1&filters=%7B%22dangling%22%3A%5B%22true%22
%5D%7D: dial unix /var/run/docker.sock: no such file or directory. Are you trying to connect to a
TLS-enabled daemon without TLS?
Get http:///var/run/docker.sock/v1.19/images/json?all=1: dial unix /var/run/docker.sock: no
such file or directory. Are you trying to connect to a TLS-enabled daemon without TLS?
-bash: lolcat: command not found

mary at meepers in ~
```

```
$ bash '/Applications/Docker Quickstart Terminal.app/Contents/Resources/Scripts/start.sh'
Creating Machine dev...
Creating VirtualBox VM...
Creating SSH key...
Starting VirtualBox VM...
Starting VM...
To see how to connect Docker to this machine, run: docker-machine env dev
Starting machine dev...
Setting environment variables for machine dev...

                        ##         .
                  ## ## ##        ==
               ## ## ## ## ##    ===
           /"""""""""""""""""\___/ ===
      ~~~ {~~ ~~~~ ~~~ ~~~~ ~~~ ~ /  ===- ~~~
           \______ o           __/
             \    \         __/
              \____\_______/

The Docker Quick Start Terminal is configured to use Docker with the “default” VM.
```

3. ターミナル・ウインドウをマウスでクリックし、アクティブにします。

ターミナル画面に不慣れでしたら、ここで便利な使い方を紹介します。

プロンプトとは一般的に `$` ドル記号です。このプロンプトの後にあるコマンドライン上でコマンドを入力します。コマンドライン上ではカーソルは `|` として表示されます。コマンドを入力した後は、常にリターン・キーを押します。

4. `docker run hello-world` コマンドを実行し、リターン・キーを押します。

以下のコマンドは、何らかの処理を行うものです。正常に終われば、画面には次のように表示されます。

```
$ docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
535020c3e8ad: Pull complete
af340544ed62: Pull complete
Digest: sha256:a68868bfe696c00866942e8f5ca39e3e31b79c1e50feaee4ce5e28df2f051d5c
Status: Downloaded newer image for hello-world:latest

Hello from Docker.
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker Engine CLI client contacted the Docker Engine daemon.
 2. The Docker Engine daemon pulled the "hello-world" image from the Docker Hub.
 3. The Docker Engine daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker Engine daemon streamed that output to the Docker Engine CLI client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
```

```
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker Hub account:
 https://hub.docker.com

For more examples and ideas, visit:
 https://docs.docker.com/userguide/
```

問題解決のヘルプをお探しですか？

通常、これらの手順は特に何も考えなくても実行できますが、もしかしたら問題が発生する場合があるかもしれません。 `docker run hello-world` が実行できずエラーになる場合は、一般的な問題を解決するためのトラブルシューティング[*1] をご覧ください。

NDIS6 ホスト・ネットワーク・フィルタ・ドライバの使用時は、Windows 固有の問題に遭遇するかもしれません。これは特定 Windows バージョンでの発生が判明してます。Windows Vista 以上のバージョンでは、VirtualBox が NDIS6 ドライバをデフォルトでインストールします。問題が発生する範囲は、仮想マシンの停止時にネットワークで問題が発生するかもしれません。もし問題が発生したら、Docker Toolbox インストーラを再実行し、VirtualBox を NDIS6 ドライバを一緒にインストールするようオプションをお選びください。

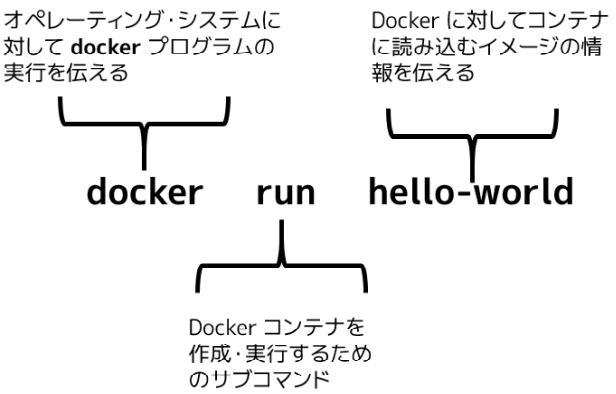
次は何をしますか

以上で Docker ソフトウェアのインストールが完了しました。Docker Quickstart ターミナル画面は開いたままにします。次はイメージとコンテナを学びますます。

*1 http://docs.docker.jp/toolbox/troubleshoot.html

1.2 イメージとコンテナを学ぶ

Docker Engine は Docker の中心となる技術を提供します。これはイメージとコンテナを扱う技術です。先ほどのステップでインストールを終えていますので、 `docker run hello-world` コマンドを実行できます。このコマンドを１つ実行するだけで、Engine を使う上で中心となるタスクをこなします。コマンドは３つのパーツに分かれています。



コンテナを基礎まで剥がしていくと、そこには Linux オペレーティング・システムがあります。イメージとはコンテナ内に積み込むソフトウェアです。コマンドを実行したとき、Engine ソフトウェアは次の処理をしました。

- `hello-world` ソフトウェアのイメージを持っているかどうかを確認。
- Docker Hub （詳しくは後述）からイメージをダウンロード。
- コンテナにイメージを載せて（読み込んで）「実行」。

イメージが何を実行するかは、イメージがどのように構築されたかに依ります。このイメージは `Hello-World` を表示するという単純なコマンドを持ちます。

Docker イメージ次第で様々な処理ができます。データベースのような複雑なソフトウェアでも、イメージを使って実行できます。あなた（もしくは誰かが）データを追加するのに待つ必要はなく、保管したデータをすぐに使えます。さらに、次の人もすぐに使えるのです。

さて、`hello-world` ソフトウェアのイメージは、誰が作ったのでしょうか。このイメージは Docker が作りましたが、イメージそのものは誰でも作れます。Docker Engine は人々（あるいは会社）が作成したソフトウェアを、Docker イメージを通して共有できるようにします。 Docker イメージ内のソフトウェアをどのコンピュータで実行すべきか、Docker Engine を使えば迷う必要がありません。なぜなら、Docker コンテナをどこでも実行できるからです。

次は何をしますか

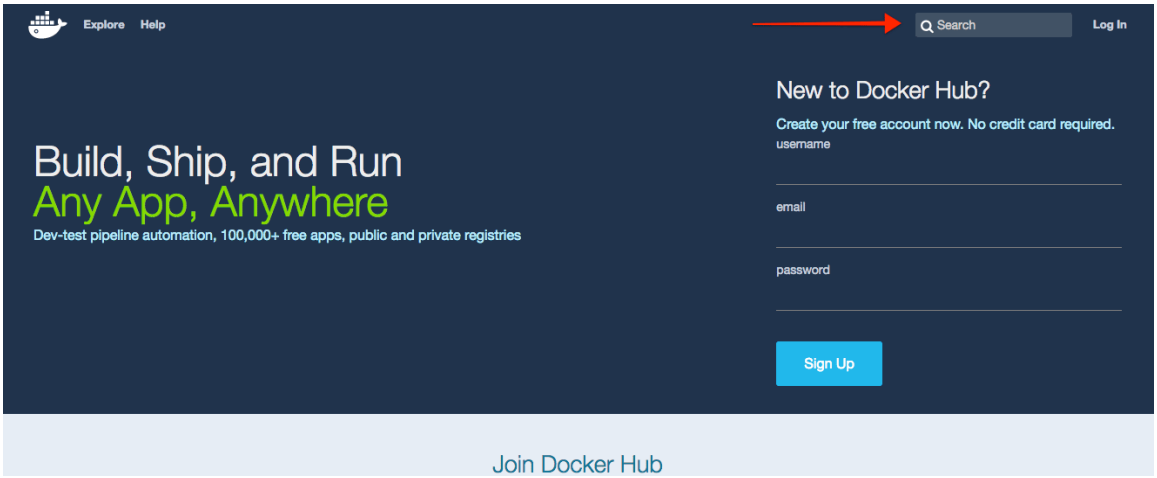
ご覧の通り、迅速だとは思いませんか。これで Docker を使って面白いことをする準備が整いました。次は whalesay イメージの実行に進みましょう。

1.3 whalesay イメージの検索と実行

世界中の皆さんが Docker イメージを作成しています。公開されたイメージは Docker Hub 上で閲覧できます。このセクションではイメージを探し出し、そのイメージを使い始めましょう。

ステップ 1 ：whalesay イメージを探す

1. ブラウザを起動し、 Docker Hub （https://hub.docker.com/）を開きます 。



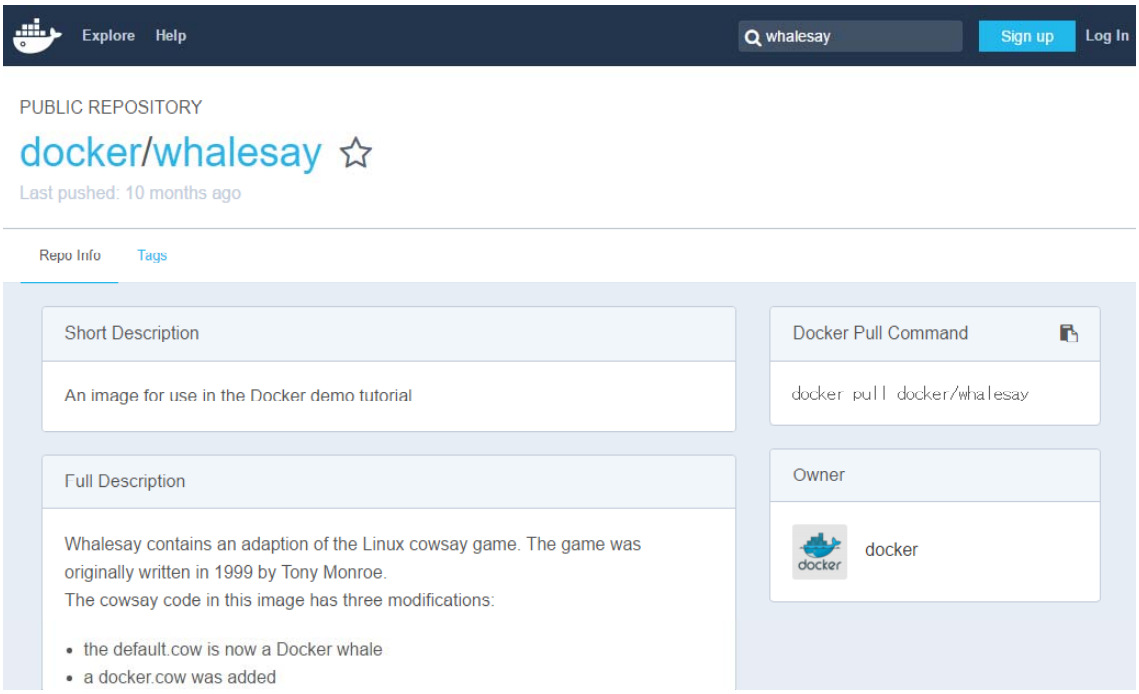
Docker Hub には皆さんのような個人で作成したイメージと、何らかの組織、例えば Red Hat、IBM、Google 等が作成した公式イメージ（オフィシャル・イメージ）があります。

2. 検索フォーム［ Search ］をクリックします。

3. 検索バーに whalesay（ホエールセイ） と入力します（訳者注： whalesay とは whale say = 鯨が話す、という意味です)。

ブラウザは whalesay イメージのリポジトリ[*1] を表示します。

*1 リポジトリ(repository)とは倉庫の意味です。Docker ではイメージを置いてある場所のことをリポジトリと呼びます。



各イメージのリポジトリにはイメージに関する情報を掲載しています。この中には、イメージにどのような種類のソフトウェアが入っているかや、使い方の説明があるでしょう。この段階で覚えておくのは、whalesay イメージとは Ubuntu と呼ばれる Linux ディストリビューションをベースにしている点です。次のステップでは、自分のマシン上で whalesay イメージを実行しましょう。

ステップ２：whaysay イメージの実行

Docker クイックスタート・ターミナルを開いていなければ、今から開きます。

1. Launchpad を開き Docker Quickstart Terminal アイコンを探します。

2. アイコンをクリックし、Docker クイックスタート・ターミナルを起動します。

3. Docker クイックスタート・ターミナルの $ プロンプトにカーソルを移動します。

4. `docker run docker/whalesay cowsay boo`（ドッカー ラン ドッカー ホエールセイ カウセイ ブー） コマンドを入力し、リターンキーを押します。

これはコンテナ内の whalesay イメージにあるコマンドを実行します。ターミナル上では、次のように表示されるでしょう。

```
$ docker run docker/whalesay cowsay boo
Unable to find image 'docker/whalesay:latest' locally
latest: Pulling from docker/whalesay
e9e06b06e14c: Pull complete
a82efea989f9: Pull complete
37bea4ee0c81: Pull complete
07f8e8c5e660: Pull complete
676c4a1897e6: Pull complete
5b74edbcaa5b: Pull complete
1722f41ddcb5: Pull complete
99da72cfe067: Pull complete
5d5bd9951e26: Pull complete
```

```
fb434121fc77: Already exists
Digest: sha256:d6ee73f978a366cf97974115abe9c4099ed59c6f75c23d03c64446bb9cd49163
Status: Downloaded newer image for docker/whalesay:latest
 _____
< boo >
 -----
    \
     \
      \
                    ##        .
              ## ## ##       ==
           ## ## ## ##      ===
       /""""""""""""""""___/ ===
  ~~~ {~~ ~~~~ ~~~ ~~~~ ~~ ~ /  ===- ~~~
       \______ o          __/
        \    \        __/
          \____\______/
```

docker コマンドを手元（ローカル）のシステム上でソフトウェア・イメージを初めて実行しました。イメージが手元になければ、 docker は Docker Hub から取得します。

5. ターミナルを開いたまま docker images コマンドを入力し、リターンキーを押します。

このコマンドは手元のシステム上にある全イメージを表示します。イメージの一覧に docker/whalesay イメージが見えるでしょう。

```
$ docker images
REPOSITORY          TAG         IMAGE ID            CREATED             VIRTUAL SIZE
docker/whalesay     latest      fb434121fc77        3 hours ago         247 MB
hello-world         latest      91c95931e552        5 weeks ago         910 B
```

コンテナ内でイメージの実行時、Docker は手元のコンピュータ上にイメージをダウンロードします。イメージのコピーを手元に作成するため、以後の作業で時間を節約します。Docker が再びイメージをダウンロードするのは、 Docker Hub 上の元イメージに変更が加わった時のみです。もちろん、イメージは自分で削除もできます。詳細は後ほど学びます。この後でもイメージを使うため、今はこのままにしておきます。

6. もう少し whalesay コンテナで遊んでみましょう。

whalesay イメージを再度使いますが、今度は言葉を換えてみましょう。長い、もしくは短いフレーズに置き換えます。何かを話せたでしょうか。

```
$ docker run docker/whalesay cowsay boo-boo
 _________
< boo-boo >
 ---------
    \
     \
      \
                    ##        .
              ## ## ##       ==
           ## ## ## ##      ===
       /""""""""""""""""___/ ===
  ~~~ {~~ ~~~~ ~~~ ~~~~ ~~ ~ /  ===- ~~~
       \______ o          __/
        \    \        __/
          \____\______/
```

次は何をしますか

このセクションでは Docker Hub 上のイメージを探す方法を学びました。コマンドを使ってイメージを実行しました。そして、自分のコンピュータ上にイメージをコピーし、実行する方法を学びました。次は自分で Docker イメージを作りましょう。

1.4 自分でイメージを構築

whalesay イメージを更に改良できます。もしかすると、何も喋らせたくないかもしれません。あるいは、もっと喋らせることもできます。

```
docker run docker/whalesay cowsay boo-boo
```

このセクションでは whalesay イメージを改良します。(オプションを指定しなくても)「自分で何かを喋る」新しいバージョンのイメージを作成します。実行に必要となるのは、ほんの少しの単語です。

ステップ１：Docker クイックスタート・ターミナルを開く

ターミナルを開いていなければ、新しいものを開きます。

1. Launchpad を起動し、Docker Quickstart Terminal のアイコンを探します。

2. アイコンをクリックし、Docker クイックスタート・ターミナルを起動します。

3. Docker クイックスタート・ターミナルのプロンプトにカーソルを合わせます。

4. 新しいディレクトリを作成するため、 mkdir mydockerbuild を入力してリターンキーを押します。

```
$ mkdir mydockerbuild
```

このディレクトリは構築時の「コンテクスト」(context；内容物）としての役割があります。このコンテクストとは、イメージを構築するために必要な全てを指します。

5. 新しいディレクトリに移動します。

```
$ cd mydockerbuild
```

この時点でディレクトリには何もありません。

6. touch(タッチ) Dockerfile と入力してエンターキーを押すと、現在のディレクトリに Dockerfile を作成します。

```
$ touch Dockerfile
```

コマンドを実行しても何も表示されませんが、実際には現在のディレクトリ内に Dockerfile が作成されています。 ls Dockerfile を実行してみましょう。

```
$ ls Dockerfile
Dockerfile
```

7. 次は open -e Dockerfile を実行し、Mac のテキストエディット・プログラムを開きます。

Mac はテキストエディット・プログラムを実行し、空の Dockerfile を開きます。

8. 開いたファイルの中に FROM docker/whalesay:latest と入力します。

次のような表示になります。

```
FROM docker/whalesay:latest
```

FROM（フロム） キーワードは Docker に対してイメージの元となるイメージを伝えます。これから作成する新しいイメージは、既存の whalesay イメージを使います。

2. 次はイメージに fortunes（フォーチューンズ） プログラムを追加します。

```
FROM docker/whalesay:latest

RUN apt-get -y update && apt-get install -y fortunes
```

fortunes プログラムは賢そうな言葉を表示するプログラムです。これを今回のこの鯨プログラムに喋らせます。そのための最初のステップは、ソフトウェアのインストールです。この行はイメージ内にソフトウェアをインストールします。

10. イメージに必要なソフトウェアをインストールしたら、イメージの読み込み時に実行するソフトウェアを命令します。

```
FROM docker/whalesay:latest

RUN apt-get -y update && apt-get install -y fortunes

CMD /usr/games/fortune -a | cowsay
```

この行は fortune プログラム（の結果）を、気の利いたことを喋る cowsay プログラムに送ります。

11. 編集した Dockerfile プログラムを保存します。メモ帳のメニューで［ ファイル(F) ］→［ 上書き保存(S) ］を選びます。

以上で Dockerfile 中にソフトウェア全ての要素と挙動を記述しました。これで新しいイメージを構築する準備が整いました。

ステップ２：Dockerfile を使ってイメージ構築

1. Docker クイックスタート・ターミナルにカーソルを合わせます。

2. Dockerfile が正確かどうかを確認するため、現在のディレクトリで cat Dockerfile を実行します。

```
$ cat Dockerfile
FROM docker/whalesay:latest

RUN apt-get -y update && apt-get install -y fortunes

CMD /usr/games/fortune -a | cowsay
```

3. 次は新しいイメージを構築するため docker build -t docker-whale . コマンドをターミナル上で実行します（最後のピリオド . を忘れないでください）。

```
$ docker build -t docker-whale .
Sending build context to Docker daemon 158.8 MB
...省略...
Removing intermediate container a8e6faa88df3
Successfully built 7d9495d03763
```

このコマンドを実行後、結果が出るまで数秒ほどかかります。この新しいイメージを使う前に、Dockerfile 構築時の流れを学びましょう。

ステップ３：構築時の流れを学ぶ

docker build -t docker-whale . コマンドは、現在のディレクトリ内にある Dockerfile を使います。そして、自分のマシン上に docker-whale という名称のイメージを構築します。コマンドの処理には少し時間がかかります。処理結果の表示は少し複雑に見えるでしょう。このセクションでは、各メッセージの意味を学びます。

まず Docker は構築時に必要な全てを確認します。

```
Sending build context to Docker daemon 158.8 MB
```

それから Docker は whalesay イメージを読み込みます。読み込むイメージは、先ほどのステップで既にローカルにあります。そのため、Docker は改めてダウンロードしません。

```
Step 0 : FROM docker/whalesay:latest
 ---> fb434121fc77
```

Docker は次の行に移ります。 apt-get パッケージ・マネージャを更新します。ここでは多くのメッセージが表示されますが、表示されるのは初回だけです。

```
Step 1 : RUN apt-get -y update && apt-get install -y fortunes
 ---> Running in 27d224dfa5b2
Ign http://archive.ubuntu.com trusty InRelease
Ign http://archive.ubuntu.com trusty-updates InRelease
Ign http://archive.ubuntu.com trusty-security InRelease
Hit http://archive.ubuntu.com trusty Release.gpg
....snip...
Get:15 http://archive.ubuntu.com trusty-security/restricted amd64 Packages [14.8 kB]
Get:16 http://archive.ubuntu.com trusty-security/universe amd64 Packages [134 kB]
Reading package lists...
---> eb06e47a01d2
```

それから、Docker は新しい fortunes ソフトウェアをインストールします。

```
Removing intermediate container e2a84b5f390f
Step 2 : RUN apt-get install -y fortunes
 ---> Running in 23aa52c1897c
Reading package lists...
Building dependency tree...
Reading state information...
```

```
The following extra packages will be installed:
  fortune-mod fortunes-min librecode0
Suggested packages:
  x11-utils bsdmainutils
The following NEW packages will be installed:
  fortune-mod fortunes fortunes-min librecode0
0 upgraded, 4 newly installed, 0 to remove and 3 not upgraded.
Need to get 1961 kB of archives.
After this operation, 4817 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu/ trusty/main librecode0 amd64 3.6-21 [771 kB]
...snip......
Setting up fortunes (1:1.99.1-7) ...
Processing triggers for libc-bin (2.19-0ubuntu6.6) ...
 ---> c81071adeeb5
Removing intermediate container 23aa52c1897c
```

最後に Docker は構築終了を画面に表示します。

```
Step 3 : CMD /usr/games/fortune -a | cowsay
 ---> Running in a8e6faa88df3
 ---> 7d9495d03763
Removing intermediate container a8e6faa88df3
Successfully built 7d9495d03763
```

ステップ４：新しい docker-whale を実行

このステップではコンピュータ上にイメージがあるかどうか確認してから、新しいイメージを実行します。

1. ターミナル・ウインドウ上でなければ、Docker クイックスタート・ターミナル画面にカーソルを合わせます。

2. docker images（イメージーズ） を実行してリターンキーを押します。

このコマンドはローカルにあるイメージの一覧を表示します。覚えておくと良いでしょう。

```
$ docker images
REPOSITORY          TAG         IMAGE ID         CREATED             VIRTUAL SIZE
docker-whale        latest      7d9495d03763     4 minutes ago       273.7 MB
docker/whalesay     latest      fb434121fc77     4 hours ago         247 MB
hello-world         latest      91c95931e552     5 weeks ago         910 B
```

3. 新しいイメージを実行します。docker run docker-whale を入力して、エンターキーを押します。

```
$ docker run docker-whale

 ----------------------------------------
/ "He was a modest, good-humored boy. It  \
\ was Oxford that made him insufferable." /
 ----------------------------------------
        \
         \
          \
                      ##        .
                ## ## ##       ==
             ## ## ## ##      ===
         /""""""""""""""""___/ ===
    ~~~ {~~ ~~~~ ~~~ ~~~~ ~~ ~ /  ===- ~~~
         \______ o          __/
          \    \        __/
            \____\______/
```

ご覧の通り、少し賢くなった鯨プログラムを作りました。コマンドラインで何かを自分で指定すると、その表示もできます！ そして、Docker は何もダウンロードしないことにも注目します。これはイメージをローカルで構築しており、ダウンロードする必要がないからです。

次は何をしますか

このセクションでは自分で Dockerfile　を記述し、イメージを構築する方法を学びました。そして、自分のイメージを使ってコンテナを実行しました。また、Mac 上の Linux システムを使い続けています。次のセクションではイメージを共有する第一歩として、 Docker Hub アカウントを作成します。

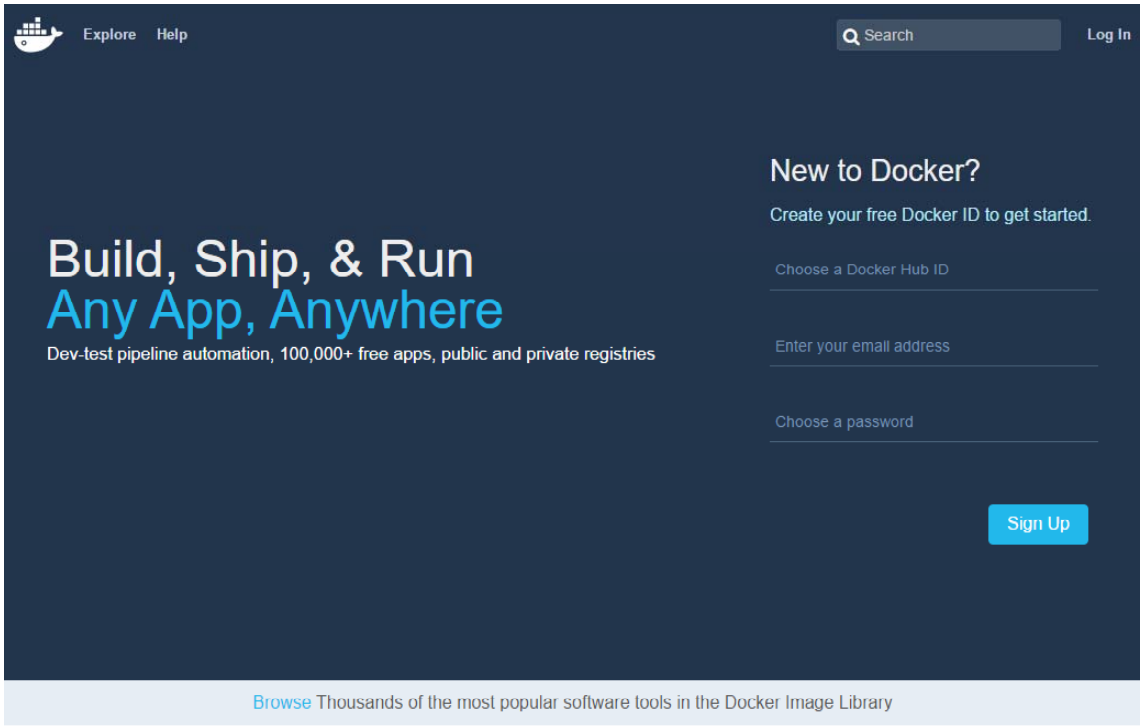
1.5 DockerHubアカウントとリポジトリ作成

良い感じにイメージを構築できました。次はイメージを共有しましょう。このセクションではイメージ共有を扱います。共有するには Docker Hub アカウントが必要です。イメージをアップロードしたら、Docker Engine を持つ他の人も実行できるようになります。

ステップ１：アカウントの登録（サインアップ）

1. ブラウザで Docker Hub のサインアップ・ページ（https://hub.docker.com/）を開きます。

ブラウザには次のページが表示されます。



The most innovative companies use Docker

2. サインアップ・ページのフォームを入力します。

Docker Hub は無料で使えます。登録にはユーザ名、パスワード、メールアドレスが必要です。

3. ［ Signup ］（サインアップ）を押します。

ブラウザには Docker Hub のウェルカム・ページが表示されます。

ステップ２：メールアドレスの確認とリポジトリの追加

何かを Docker Hub 上で共有する前に、メールアドレスの確認が必要です。

1. メールの受信箱を確認します。

2. メールのタイトル `Please confirm email for your Docker Hub account` （Docker Hub アカウントのメールアドレスをご確認ください）を探します。

メールが届いていなければ、迷惑メール用のフォルダに入っていないか確認するか、メールが到着するまでお待ちください。

3. メールの本文にあるボタンをクリックし、メールアドレスの確認を済ませます。

ブラウザで Docker Hub のページを開くと、自分のプロフィール・ページを表示します。

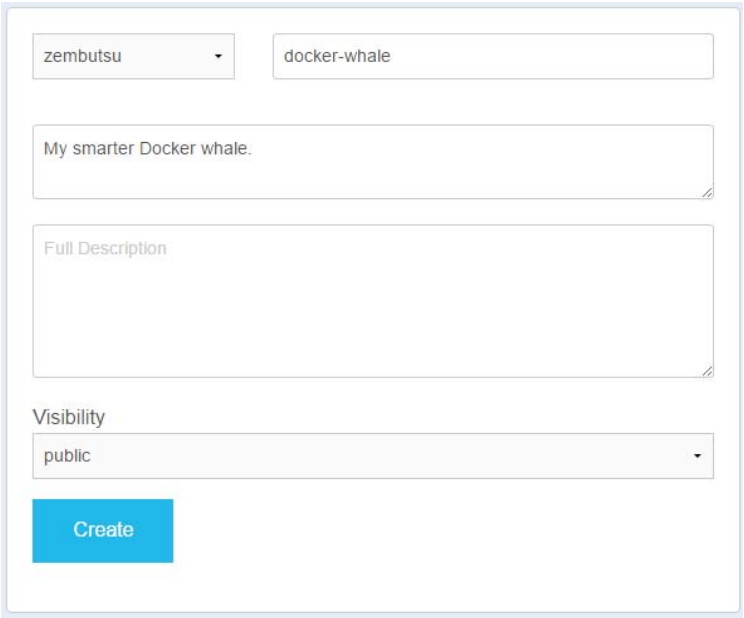
4. ［ Create（クリエイト） Repository（リポジトリ） ］（リポジトリの作成）をクリックします。

ブラウザは新しいリポジトリ作成ページを開きます。

5. リポジトリ名と簡単な説明を背追加します。

6. Visibility（ビシビリティ）（可視性）は Public（パブリック） （公開）に指定します。

作成するには、次の画面のように入力します。



7. Create を押すと作業完了です。

Docker Hub に自分の新しいリポジトリを作成しました。

次は何をしますか

このセクションでは Docker Hub 上にアカウントを開設し、新しいリポジトリを追加しました。次のセクションでは、 先ほど作成したイメージにタグを付けてからリポジトリに送信します。

1.6 イメージのタグ付け、送信、取得

このセクションでは docker-whale イメージにタグを付け、先ほど作成した自分のリポジトリにイメージを送信します。作業が後は、自分の新しいイメージをリポジトリから取得できるか確認します。

ステップ１：イメージのタグ付けと送信

Docker クイックスタート・ターミナルを開いていなければ、今から開きます。

1. Launchpad を開き、Docker Quickstart Terminal アイコンを探します。

2. アイコンをクリックし、Docker クイックスタート・ターミナルを起動します。

3. Docker クイックスタート・ターミナル画面のプロンプトに、カーソルを合わせます。

4. docker images を入力し、手元に現在あるイメージの一覧を表示します。

```
$ docker images
REPOSITORY          TAG         IMAGE ID          CREATED             VIRTUAL SIZE
docker-whale        latest      7d9495d03763       38 minutes ago      273.7 MB
<none>              <none>      5dac217f722c       45 minutes ago      273.7 MB
docker/whalesay     latest      fb434121fc77       4 hours ago         247 MB
hello-world         latest      91c95931e552       5 weeks ago         910 B
```

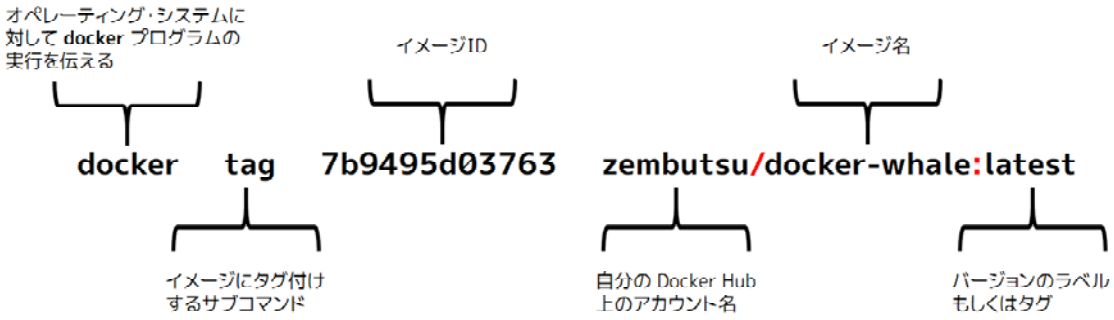
5. docker-whale イメージの IMAGE ID を確認します。

この例の ID は 7d9495d03763 です。

現時点の REPOSITORY にあるリポジトリ名は docker-whale ですが、名前空間（namespace）がありません。名前空間は Docker Hub 上の自分のアカウントに関連付いています。この名前空間は自分の Docker Hub アカウント名と同じです。そのため、イメージを 自分のアカウント名/docker-whale に名称変更する必要があります。

6. docker tag（タグ） コマンドと イメージ ID を指定し、 docker-whale イメージをタグ付けします。

入力するコマンドには、次の意味があります。



もちろん、アカウント名は自分自身のものです。そのため、イメージ ID やアカウント名は自分のものを入力し、リターンキーを押します。

```
$ docker tag 7d9495d03763 maryatdocker/docker-whale:latest
```

7. docker images コマンドをもう一度実行して、新しくタグ付けされたイメージがあるかどうか確認します。

```
$ docker images
REPOSITORY                  TAG       IMAGE ID        CREATED          VIRTUAL SIZE
maryatdocker/docker-whale   latest    7d9495d03763    5 minutes ago    273.7 MB
docker-whale                latest    7d9495d03763    2 hours ago      273.7 MB
<none>                      <none>    5dac217f722c    5 hours ago      273.7 MB
docker/whalesay             latest    fb434121fc77    5 hours ago      247 MB
hello-world                 latest    91c95931e552    5 weeks ago      910 B
```

8. コマンドライン上で docker login（ログイン） コマンドを使い Docker Hub にログインします。

ログインコマンドの書式は次の通りです。

```
docker login --username=yourhubusername --email=youremail@company.com
```

入力を促すプロンプトが表示されたら、パスワードを入力してエンターを押します。実行例：

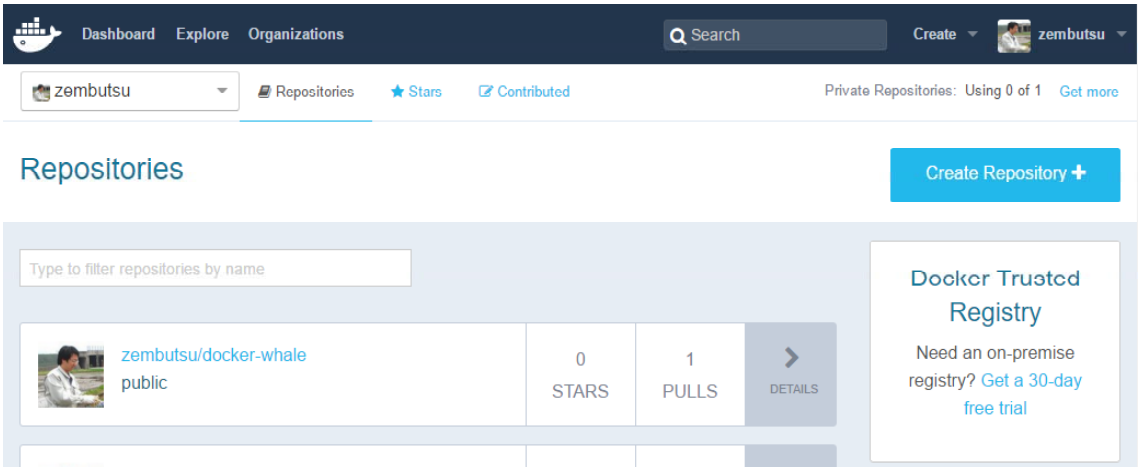
```
$ docker login --username=maryatdocker --email=mary@docker.com
Password:
WARNING: login credentials saved in C:\Users\sven\.docker\config.json
Login Succeeded
```

9. docker push（プッシュ） コマンドを実行し、自分のイメージをリポジトリに送信します。

```
$ docker push maryatdocker/docker-whale
    The push refers to a repository [maryatdocker/docker-whale] (len: 1)
    7d9495d03763: Image already exists
    c81071adeeb5: Image successfully pushed
    eb06e47a01d2: Image successfully pushed
    fb434121fc77: Image successfully pushed
    5d5bd9951e26: Image successfully pushed
    99da72cfe067: Image successfully pushed
    1722f41ddcb5: Image successfully pushed
```

```
5b74edbcaa5b: Image successfully pushed
676c4a1897e6: Image successfully pushed
07f8e8c5e660: Image successfully pushed
37bea4ee0c81: Image successfully pushed
a82efea989f9: Image successfully pushed
e9e06b06e14c: Image successfully pushed
Digest: sha256:ad89e88beb7dc73bf55d456e2c600e0a39dd6c9500d7cd8d1025626c4b985011
```

10. 自分の Docker Hub のプロフィールページに戻ります。新しいイメージの情報が表示されています。



ステップ２：新しいイメージの取得

最後のセクションでは、Docker Hub に送信(push)したイメージを取得(pull)します。作業を進める前に、これまでローカルマシン上で作成したオリジナルのイメージを削除します。マシン上にオリジナルのイメージを残しておいたままでは、Docker は Docker Hub からイメージを取得しません。これは両方のイメージが同じと認識されるためです。

1. ターミナルのウインドウ上にあるプロンプトに、カーソルを合わせます。

2. docker images を入力し、ローカルマシン上にあるイメージの一覧を表示します。

```
$ docker images
REPOSITORY                  TAG       IMAGE ID        CREATED          VIRTUAL SIZE
maryatdocker/docker-whale   latest    7d9495d03763    5 minutes ago    273.7 MB
docker-whale                latest    7d9495d03763    2 hours ago      273.7 MB
<none>                      <none>    5dac217f722c    5 hours ago      273.7 MB
docker/whalesay             latest    fb434121fc77    5 hours ago      247 MB
hello-world                 latest    91c95931e552    5 weeks ago      910 B
```

テストを正しく行うため、ローカルのシステム上から maryatdocker/docker-whale と docker-whale イメージを削除します。次の docker pull コマンドを実行する前に、リポジトリからイメージを削除します。

3. docker rmi コマンドを使い、 maryatdocker/docker-whale と docker-whale イメージを削除します。

イメージを削除するにはイメージ ID かイメージ名を使います。

```
$ docker rmi -f 7d9495d03763
$ docker rmi -f docker-whale
```

4. docker run コマンドを使い、リポジトリから新しいイメージの取得と読み込みます。

コマンド実行時、ユーザ名には Docker Hub 上の自分の名前を指定します。

```
docker run 自分のユーザ名/docker-whale
```

イメージがローカルホスト上にないため、Docker はイメージをダウンロードします。

```
$ docker run maryatdocker/docker-whale
Unable to find image 'maryatdocker/docker-whale:latest' locally
latest: Pulling from maryatdocker/docker-whale
eb06e47a01d2: Pull complete
c81071adeeb5: Pull complete
7d9495d03763: Already exists
e9e06b06e14c: Already exists
a82efea989f9: Already exists
37bea4ee0c81: Already exists
07f8e8c5e660: Already exists
676c4a1897e6: Already exists
5b74edbcaa5b: Already exists
1722f41ddcb5: Already exists
99da72cfe067: Already exists
5d5bd9951e26: Already exists
fb434121fc77: Already exists
Digest: sha256:ad89e88beb7dc73bf55d456e2c600e0a39dd6c9500d7cd8d1025626c4b985011
Status: Downloaded newer image for maryatdocker/docker-whale:latest
 ----------------------------------------
/ Having wandered helplessly into a      \
| blinding snowstorm Sam was greatly     |
| relieved to see a sturdy Saint Bernard |
| dog bounding toward him with the       |
| traditional keg of brandy strapped to  |
| his collar.                            |
|                                        |
| "At last," cried Sam, "man's best      |
\ friend -- and a great big dog, too!"   /
 ----------------------------------------
               \
                \
                 \
                        ##        .
                  ## ## ##       ==
               ## ## ## ##      ===
           /""""""""""""""""___/ ===
      ~~~ {~~ ~~~~ ~~~ ~~~~ ~~ ~ /  ===- ~~~
           \______ o          __/
            \    \        __/
              \____\______/
```

次は何をしますか

これで Docker の基本的なタスクを扱う全てが終了しました。

- Docker をインストール。
- コンテナでソフトウェアのイメージを実行。
- Docker Hub 上で興味あるイメージを探す。
- 自分のマシン上でイメージを実行。
- 実行するイメージに対する変更を加え、イメージを作成。
- Docker Hub 上のアカウントとリポジトリの作成。
- 他の人と共有できるよう、 Docker Hub にイメージを送信。

完了したことを Tweet しましょう！

Docker ができることを詳しく知りたくありませんか。

皆さんの興味に応じた、豊富な Docker のドキュメントがあります。

- Mac OS X 上で Docker Engine を使うための基本を学びたい場合。Docker Engine を更に使うため、Docker Toolbox のユーザ・ガイドを使ってインストールしたい場合。

  Docker Toolbox のインストール

  http://docs.docker.jp/engine/installation/windows.html

- Docker プロダクト郡についての情報を知りたい場合。

  各プロダクトの紹介ページ（英語）

  http://www.docker.com/products

- Docker Hub で自動構築（automated build）を使いたい場合。

  Docker Hub ドキュメント

  http://docs.docker.jp/docker-hub/index.html

- Compose で複数のコンテナを扱うアプリケーションを実行したい場合。

  Docker Compose ドキュメント

  http://docs.docker.jp/compose/gettingstarted.html