Running Containers Exercise

Goal:

To familiarize yourself with the various options for the “docker run” command.

Instructions:

Start a Container

Start up a container with a simple “`docker run`” command using the popular key-value database, Redis:

```
docker run -dit redis
```

Check to see if it is running:

```
docker ps
```

Look closely at the left-hand side for the container hash under the CONTAINER ID column.

Also look at the arbitrary name that Docker has assigned the container, as you haven’t explicitly named it. That information is on the right-hand side, under the NAMES column.

Start a Container With a Name

To stop that Redis container, you need to use its hash or arbitrary name.

Start a container with a name we can reference again later:

```
docker run -dit --name redis_container redis
```

NOTE: You can also use the "`--name=CONTAINER_NAME`" syntax.

Check that it’s running:

```
docker ps
```

Stop a Named Container

Next, stop that container you named:

```
docker stop redis_container
```

Check the running containers on the system again:

```
docker ps
```

We should only see the one Redis container, which received an arbitrary name from Docker.

Automatically deleting a stopped container

Let's run a container with the "--rm" option. This option tells Docker to automatically delete the container once it stops.

We will be using an image called "hello-world", which Docker uses for demonstrations. If you don't have a local copy of the image, Docker will download it before starting the container.

```
docker run --rm hello-world
```

Check that the image has been deleted after it has completed its execution:

```
docker ps -a
```

Now run a container using the hello-world image without the "`--rm`" option, as shown here:

```
docker run hello-world
```

Check that it has not been deleted this time:

```
docker ps -a
```

Some users of Docker frequently use the "`--rm`"option to help with garbage collection.

Logs

The “`docker logs`” command provides information directly from the STDOUT (standard output) and STDERR (standard error) of a container.

The format of the command is:

```
docker logs CONTAINER_NAME
```

Create a container using the “hello-world” image:

```
docker run --name test_container -d hello-world
```

Examine the output of the "`docker ps`" command:

```
docker ps
```

Think about the commands you just executed and answer the following questions:

1. Why isn’t the container visibly running with a “`docker ps`” command?
2. What command will show details of the container?
3. How would you read the output from that container, using the “`docker logs`” command?

The answer to the first question is because the container already stopped by the time you executed the "`docker ps`" command. The hello-world image simply outputs some information and then stops. Unlike some other containers, it doesn't provide an ongoing service.

The answer to the second question is to use "`docker ps -a`". That command allows you to view all containers, including those that are stopped.

```
docker ps -a
```

The answer to the third question is to run "`docker logs test_container`" to see the output from the container.

Use the name of the container you gave it, “test_container” instead of using the container ID (hash).

```
docker logs test_container
```

Now display the output of the container with timestamps:

```
docker logs -t test_container
```