How to run Hello World Docker Activity Plugin Unit Test.

1. Create a container by name
   Run `org.apache.taverna.activities.docker.test.DockerActivityTest > testCreateContainer()`
   Example output log

   ```json
   {"container-id":"68183dda9ad4d85927cf5a66006de63963ec06488fdf6215e7db4e645e04d123"}
   ```

2. Here is the "docker ps -a" output will look like after step #1

<table>
<thead>
<tr>
<th>CONTAINER ID</th>
<th>IMAGE</th>
<th>COMMAND</th>
<th>CREATED</th>
<th>STATUS</th>
<th>PORTS</th>
<th>NAMES</th>
</tr>
</thead>
<tbody>
<tr>
<td>68183dda9ad4</td>
<td>training/webapp</td>
<td>&quot;python app.py&quot;</td>
<td>9 minutes ago</td>
<td>Created</td>
<td></td>
<td>test-container</td>
</tr>
</tbody>
</table>

3. Option step (List all containers available)
   Run `org.apache.taverna.activities.docker.test.DockerActivityTest > testListContainers()`
   Output json array (now only one element, because we just created one container):

   ```json
   [  
     {  
       "id": "68183dda9ad4d85927cf5a66006de63963ec06488fdf6215e7db4e645e04d123",
       "command": "python app.py",
       "created": 1469947093,
       "image": "training/webapp",
       "image-id": "sha256:6f883f344649a39240b94d73b8ba9c67f898ede85c8e97a8873e6557",
       "status": "Created",
       "network-mode": "default"
     }
   ]
   ```

4. Start Container by name:
   Run `org.apache.taverna.activities.docker.test.DockerActivityTest > testStartContainer()`
   Example output log

   ```json
   {"started":"68183dda9ad4d85927cf5a66006de63963ec06488fdf6215e7db4e645e04d123"}
   ```
5. After step 4, here is how docker ps command output will look like. If you compare the column “status” with step#2 output, you can see now the container is moved from created state to Up state.

<table>
<thead>
<tr>
<th>CONTAINER ID</th>
<th>IMAGE</th>
<th>COMMAND</th>
<th>CREATED</th>
<th>STATUS</th>
<th>PORTS</th>
<th>NAMES</th>
</tr>
</thead>
<tbody>
<tr>
<td>68183dda9ad4</td>
<td>training/webapp</td>
<td>&quot;python app.py&quot;</td>
<td>21 minutes ago</td>
<td>Up 3 minutes</td>
<td>0.0.0.0:32772-&gt;5000/tcp</td>
<td>test-container</td>
</tr>
</tbody>
</table>

6. Here at step 5, we actually started docker python webapp “training/webapp”. You can actually visit the web app from browser to check the app actually running in the container with the exposed port “32772”

Hello world!

7. Let’s stop the container by name.
   Run org.apache.taverna.activities.docker.test.DockerActivityTest > testStopContainer()
   Example output log

   ```json
   {"stopped":"68183dda9ad4d85927cf5a66006de63963ec06488fdf6215e7db4e645e04d123"}
   ```

8. Now let’s see docker ps -a command output. And now it is in “Exited” status.

<table>
<thead>
<tr>
<th>CONTAINER ID</th>
<th>IMAGE</th>
<th>COMMAND</th>
<th>CREATED</th>
<th>STATUS</th>
<th>PORTS</th>
<th>NAMES</th>
</tr>
</thead>
<tbody>
<tr>
<td>68183dda9ad4</td>
<td>training/webapp</td>
<td>&quot;python app.py&quot;</td>
<td>32 minutes ago</td>
<td>Exited (137)</td>
<td>About a minute ago</td>
<td>test-container</td>
</tr>
</tbody>
</table>

8. Let’s delete the container by name

Run org.apache.taverna.activities.docker.test.DockerActivityTest > testDeleteContainer()
   Example output log

   ```json
   {"deleted":"68183dda9ad4d85927cf5a66006de63963ec06488fdf6215e7db4e645e04d123"}
   ```

9. Finally, the docker ps -a command output would be empty as expected.

| CONTAINER ID | IMAGE           | COMMAND       | CREATED         | STATUS           | PORTS                  | NAMES           |