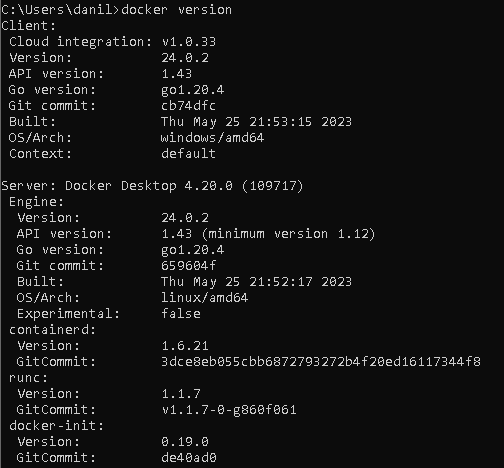
**Docker terminology  
Изображение выглядит как текст, число, Шрифт, чек

Автоматически созданное описание**

**-----------------------------------------------  
  
Docker commands:**(Pre-cond: docker client must be running)

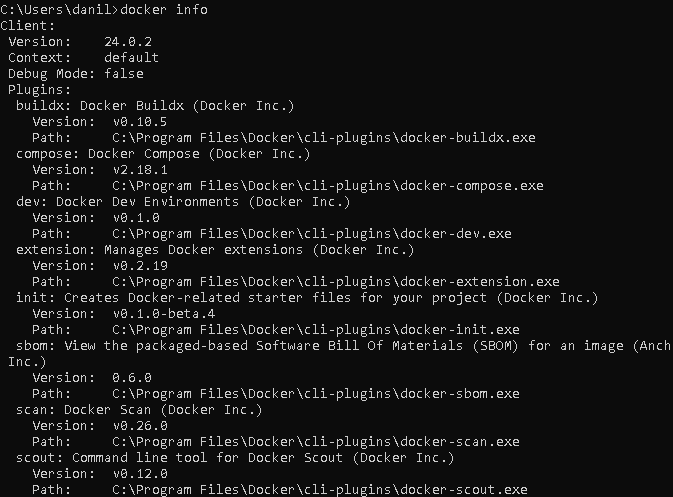
---------------  
  
docker version (gives the current info about docker both client and server)  
  
****--------------------------------------------------

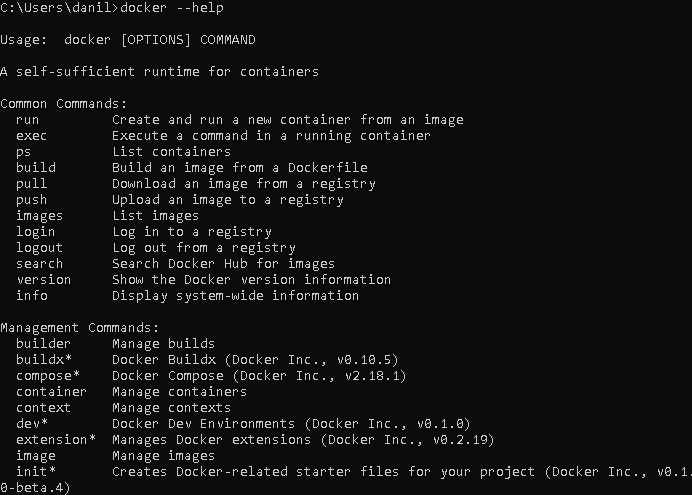
docker -v (gives only version of the docker)

docker --version (the similar command)  
  
Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

-------------------------------------

docker info (gives diverse detailed info about docker(images, containers, etc.))  
  


--------------------------------------------  
  
docker --help (generates lists of various command and description how to use them)  
  


-----------------------------  
  
docker login (you’re supposed to be registered for docker hub); after running this command – type your credentials to log in  
  
\*I was already logged in so didn’t write down login and password  
  
Изображение выглядит как текст, снимок экрана, Шрифт, черный

Автоматически созданное описание

--------------------------------  
  
docker logout (to log out from the docker hub)



---------------------------------  
  
  
  
docker images (presents all the images in my system)

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

---------------------------------------

\*Through the docker hub we can check which images are available in the opensource and complete the ‘pull’ command to download some definite image.  
  
------------------------------------------  
  
  
docker pull <image\_name> (pulling the image from docker hub to your system)

Изображение выглядит как текст, снимок экрана, Шрифт, черный

Автоматически созданное описание

-----------------------------------------  
  
docker rmi <image\_id> (by passing the id we removing this particular image)

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

-----------------------------------------------  
  
docker ps (list of containers(currently there’s no any))  
Изображение выглядит как текст, Шрифт, снимок экрана, черный

Автоматически созданное описание  
----------------------------------------------  
  
  
  
docker run <image\_name>(in some parts similar to docker pull command but with some differensies) – but docker run first will check if there’s this image in my system or not, if it’s not it’l pull this image and save it into my system  
-------------------------------------------------  
  
docker run -it <image\_name> (this command will launch and create your container via image you provided (ubuntu or another one))  
Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

------------------------------------------------  
  
docker stop <container\_id> (this command will terminate your container with this id)

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

----------------------------------------------

docker start <container\_id> (this command will commence your container with this id)  
Изображение выглядит как текст, снимок экрана, Шрифт, черный

Автоматически созданное описание

--------------------------------------------  
  
docker stats (this command will give the status of my containers with information)

Изображение выглядит как текст, снимок экрана, меню, черно-белый

Автоматически созданное описание

------------------------------------------  
  
docker system df (it’ll give the info about images containers how many of them, preoccupied size, volume and so forth)  
Изображение выглядит как текст, снимок экрана, Шрифт, черный

Автоматически созданное описание

----------------------------------------  
  
docker system prune -f (remove all stopped containers, so all of them besides running ones)

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

-----------------------------------------

**NEXT SESSION**

How to pull image from DockerHub

How to create Container

Create&Run selenium tests remotely on Container

Images:

1) Selenium StandAlone with Chrome(selenium/standalone-chrome)

2) Selenium StandAlone with FireFox(selenium/standalone-firefox)

At first I pull the image:

Изображение выглядит как текст, снимок экрана, Шрифт, черно-белый

Автоматически созданное описание

And According to script given in the docker hub we create container from this image:  
Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

So now by using Eclipse IDE, I’ll create a simple script to run it into container:  
  
here I create a new package and new java class within a maven project.  
  
and also I added essential dependency inside pom.xml file  
  
Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

And then I wrote some lines of simple code and execute it on the remote localhost(container):

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

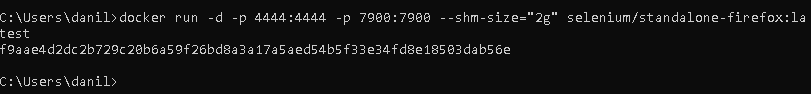
Thus, I pulled an image from the dockerhub, and I launched the container by using documentation of this particular image (selenium/standalone-chrome:latest)  
  
2) Now I’d like to do the same thing but via another browser(firefox). Not nesseceraly but I’ll stop my first container, although it’s possible to run them simultaneously.

And I’ll download another image

Изображение выглядит как текст, снимок экрана, Шрифт, дизайн

Автоматически созданное описание

And then I’ll run new container by using just received image:



Then I again open my code I created lately for chrome configuration, and I need to change ChromeOptions to FirefoxOptions and run it:

Изображение выглядит как текст, электроника, снимок экрана, Шрифт

Автоматически созданное описание

Everything’s working

-----------------------------------------

**Selenium Grid**

Изображение выглядит как текст, компьютер, снимок экрана, программное обеспечение

Автоматически созданное описание

This is representation of selenium grid, so we have a little program which may trigger the Hub Server, located on VM1, and according to delivered configuration Hub will ask to execute out test on some particular node (VM2 Windows, VM3 Mac, VM4 Linux) on a certain browser, where VM2, VM3, VM4 are nodes or sleeved. But in our cases we’re gonna use docker containers instead of VMs.

Here's helpful documentation with various ways how to run the containers through the existed images:

<https://github.com/SeleniumHQ/docker-selenium>  
  
First approach – docker networking commands

Second approach – using docker compose (file mentioning all the configurations) – most simple approach at my point of view

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

The name of the file here’s docker-compose-v3.yml. So after execution this file with some adjustments implemented by me we’ll get all the images first and based on them all the containers will be created and our environment will be established.

And after creation of this file we’ll execute command docker-compose up from the same directory.

I’m gonna use this approach. Let’s walk through the whole process step by step:  
  
1) Create a file docker-compose.yaml with required and reconfigured settings proposed by (<https://github.com/SeleniumHQ/docker-selenium>)  
  
  
Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание  
  
  
  
  
2) Run ‘docker-compose up ‘ command from the same directory ------> that’ll establish grid environment, pull needed images and create containers (grid hub and greed nodes(linux+firefox, linux+chrome))

Изображение выглядит как текст, снимок экрана, Шрифт, черный

Автоматически созданное описание  
  
3) So after direction to the link address: <http://localhost:4444> we can observe our selenium grid environment with comprised grid hub and grid nodes:

Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

Автоматически созданное описание

4) But suppose I want to increase the amount of my chrome nodes. What I can do is to execute one more command in the command prompt: ‘docker-compose scale chrome=3’, so overall I’ll have 3 chrome nodes, so 2 appended.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Автоматически созданное описание

Now I’ll go to eclipse IDE and create a piece of code to conduct this code(test) on remote server (container – one of the grid nodes). But it’s not that simple. My requirement is to conduct 4 tests simultaneously, so hub will distribute all the tests for available nodes (that’s the main con of selenium grid)

I already mentioned pom.xml file configurations, and it’s prepared with earlier added dependency for selenium web automation testing. So I also created 4 tests (3 for chrome, 1 for firefox), they look nearly similar:

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

So I gathered all the test cases into testing.xml file and added one more setting inside the suite tag: parallel=”classes”, to make them run in parallel.

Изображение выглядит как текст, электроника, снимок экрана, Шрифт

Автоматически созданное описание

All the tests have been completed successfully:  
**\*\*\***(need to add some screens after figuring out what was the problem – Repeat -> DockerProj)

5) Finally, if I want to shut down all my containers I can execute ‘docker-compose down’ command from the same directory.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

-------------------------------------------------------

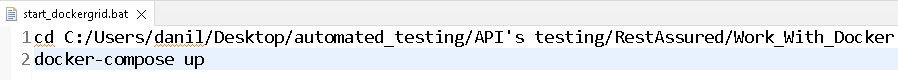
**How to invoke Selenium Grid on Docker Environment using Batch file**

First of all, I replace all into one project using Eclipse IDE:   
I’ll have my .java files representing the tests, launching on diverse containers with different configurations.  
I’ll move my docker-compose.yaml file to the project in case not to lose it in some directories. And of course I’ll have testing.xml file for running all the tests simultaneously as I made it in the previous topic.  
  
Inside this particular project I’m gonna start creating batch files filling with certain code assigning to different activities. Here comes current situation within my project:

Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Автоматически созданное описание

My first batch file is called ‘start\_dockergrid.bat’ to start docker-compose.yaml file with predefined settings for downloading images and beginning the containers based on those images.

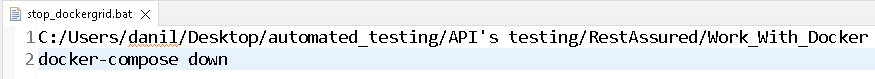


Изображение выглядит как текст, число, Шрифт, программное обеспечение

Автоматически созданное описание

So to execute this batch file we have to just double-click on it.

Now I’ll create a file to stop all the activities, to simply shut down the whole process of up and running all the containers being within grid environment. Name of the file is ‘stop\_dockergrid.bat’.



Изображение выглядит как текст, число, Шрифт, программное обеспечение

Автоматически созданное описание

We have to provide double-clicking as well to run this file.

Now I’ll launch start\_dockergrid.bat file manually, it’ll create my grid environment with grid hub and grid nodes:

Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Автоматически созданное описание

Now I’ll execute my tests again through Eclipse IDE on the remote grid nodes (containers)  
After successfully completed execution I’ll shut down my grid environment by manually launching stop\_dockergrid.bat file

It finished the process successfully.

Now I’m gonna write Java Code and from this script I’ll demonstrate opening and closing my grid environment.



After running this script by TestNg method I checked if my grid environment successfully launched and stopped.

Now I should provide such case when my program starts grid environment executes all my tests remotely and then shuts down the environment. So, instead of @Test annotations I’ll assign @Before annotation to start method and @After annotation to stop method and also I need to put one line to testing.xml file to complete this annotations in time.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Now I can run this code exactly from this place, from testing.xml file. And Everything(all the tests) has been executed successfully with automatic start and stop of grid environment.

I can also execute directly from pom.xml. But I have to add this plugins to the file.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

And I also can run this from Jenkins tool.