*Docker*

**Docker for the Absolute Beginner – Hands On - DevOps**

(Tutorial – Channel Name: Udemy)

Videos - #28 - #53

**First Published On: 15th Jun 2021**

**Last Updated On: 16th Jun 2021**

**Docker**

|  |  |
| --- | --- |
| **Watch Video Example Voting Application (Udemy Video #30)**  **Example Voting Application Using Docker Compose (Udemy Video #31) –** *watch above video #30 first to understand voting application clearly, this section is used to avoid the long process of installing and configuring connectors individually*                  **Docker Compose Latest Version, version 3 – (Udemy Video #32)**    **Docker Registry**    **Docker Engine (Udemy Video #36)**            **Docker Storage**                **Changing the files in image**        **Watch Video – Docker on Windows (Udemy Video #43)**  **Demo – Docker on Windows**                  **Docker Orchestration** | * Used to run a whole stack on single docker host * Consider, we have installed all the containers having its services and also provided names * But, now, if you want the flow from 1 container to another, it will not work as container does not know which other container to link * Thus, we will use *--links* to link the applications/services to another * *redis:redis* – means container:host * Similarly, link to db server * Similarly for worker container, but in this case, the worker is linked to both db as shown * Create a docker-compose.yml file to map all your parameters in a single file * It’s not necessary to pull image from dockerhub repo, if you have your own docker images available, you can easily use by specifying the path to your directory * You have readymade docker templates/images of OSs configured and installed and are available on dockerhub or docker store * You can just install them and run docker run <<application>> easily. * As discussed earlier, docker compose is used to run a stack of application and services * By default, docker compose is not installed, install it using shown commands * Create a docker-compose.yml file * Edit it * Give the appropriate parameters as shown * This command will run the containers as per the file * Result * Observe that first 2 lines are added * environment is added because the postgres requires username and password * Also observe, the link tag is now removed, link tag is depreciated in newer versions * The linking is done automatically * When we run container by giving name e.g docker run nginx, the docker takes it from user/repository * However, we if we just give only single name e.g. nginx, docker assumes that the user and repository names are the same * Also, the images if not found locally are pulled from Registry which is by default docker.io. * The server on which docker is installed is called as docker host * 3 components are installed as shown * Communication from CLI to Deamon happens through REST API * Docker CLI can also be installed remotely to execute operations from remote laptop * Eg, to install nginx on remote Docker Engine execute command as shown by replacing the IP and Ports * To restrict the container to use only limited resources of underlying host, use commands as shown * .5 means to use only 50% of CPU * *docker exec <<container ID>>* is used to execute command in the container * *ps -eaf* is used to get the processes running * Any process running in container also runs on underlying host * But the difference is that these two are having different PIDs (Process IDs) * Layered architecture, which means, when an image is built from dockerfile, it installs all the dependencies and files required. * However, if you use the same image to build another container, it will reuse the installation files and dependencies from the cache because they were already installed earlier * Container layer is separated than the Image Layer * Image layer is Read only and is shared among all the containers * You can mount volume to save the copy of application data e.g database data so that if the db container is destroyed, you can run the db container again using same volume path, it will restore all the data back * The correct and newer way is by using --mount * source = location on host * target = location on container * docker info shows the information of installed docker * Use this command to know the steps used by docker while installing an image * Let’s change the source code of files and also the dockerfile and build a new image * Use the newly created dockerfile by *-f* parameter * Created a new image “sample-webapp2” * Type Docker for windows on google and go to the official docker store website. * Download and install CE version * Initially Hyper-V is not enabled * Click Ok to automatically install * Once restarted, wait for some time to initialize the docker * Type docker version command to check * By default, the docker for windows runs with Linux containers * Windows has created and installed virtual Linux image. * To check this, goto Server Manager * Observe that the installed linux distro is MobyLinux * To switch to Windows containers, we can do it from taskbar right click on icon and clicking the option as shown * This is available only for Windows Server 2016+ and Windows 10 professional edition * Orchestration is concept to Scale in, Scale out the hosts/resources, monitor health and manage load balancing |