Stopped VIDEO at 1.13 minutes
Labs: kodekloud.com/p/docker-labs Kloudjvh0412

Containerize Java Boot Spring Application Steps

1) Navigate to directory containing the Java Boot Spring application	Navigate to the PC directory containing the Eclipse project: C:\Users\jvhdb\eclipse-workspace\SpringBoot-302
2) Compile Application into Jar File using maven comand. It will create an executable .jar file in the /target directory within the application directory.	nvm package jar file will end with ...SNAPSHOT.jar
3) Within application directory create the Dockerfile (no extensions)	FROM paulosalgado/oracle-java8-ubuntu-16 COPY /target/SpringBoot_302-0.0.1-SNAPSHOT.jar SpringBoot_302-0.0.1-SNAPSHOT.jar CMD java -jar SpringBoot_302-0.0.1-SNAPSHOT.jar
4) Containerize it with docker command - create docker image	docker build . -t *image_name*
5) Run as a container with docker command - deploy it	docker run -it -p 80:8080 *image_name* **-d (=detach)**

Kubernetes CLI Commands using kubectl

Command Name	Description
kubectl get nodes	displays kubernetes name and status , role and version
kubectl config view	displays kubectl configuration file displaying names of defines clusters, the users and the contexts (grouping of clusters and users)
kubectl config current-context	shows the current context
kubectl config use-context *context name*	default context name for desktop platform: docker-desktop this command sets the current context

kubectl apply -f pod.yml	cmd posts manifest file (*.yml) to the API server in master node so the container specified in the manifest file is deployed within the pod named hello-pod as specified in the yml file.
kubectl get pods	display the status of the pods
kubectl get pods *podname -o yaml*	displays details about specified pod manifest file
kubectl get pods *podname -o wide*	more cols less info than yaml
kubectl describe pods *hello-pod*	well formatted view of pod
kubectl exec hello-pod ps -aux	use exec to log into pod and run cmd
docker commands	Community Edition (free) Enterprise Edition($$)
docker version	displays the installed version of docker example: version 19.03.12
docker image ls	list available images (like un-instantiated classes) not yet started status=stopped (these are containers not yet started)
docker ps docker ps -a	shows containers name, status, id, ports.img (active only) shows all containers running and not
docker container ls	list active or running containers
docker inspect *container_name*	provides details about the container
docker logs *container_name*	will display the container generated logs
docker *command* - - help	help for specified command
docker help docker [options] *command*	generic help, command list by type (mgmt/other) also syntax including options

dockerhub docker run ansible docker run mongodb https://hub.docker.com docker run ubuntu docker run ubuntu sleep 5 (in this case sleep is a process or task)	a registry of containerized docker apps like mongodb, ansible, Use cmd docker run to instantiate or run one of these app containers using docker’s run engine running an OS like ubuntu, process will end almos immediately since containers not designed to run op systems but processes.
image	a package or a template of a container. You must “instantiate” the image to use it as a container comparable to a class and an instance of a class
docker file provided to us by Nigel Poulton in his gitHub repository /webapp/edge or /webapp/latest there is a docker file	created by developers, then used by them to create an image which can then be deployed by docket as a container
docker pull openjdk cmd displayed when selecting the image. Just copy/paste to docker cli to start	This cmd downloads (pulls) an image of the supplied application images in docker. In this case openjdk refers to the java jdk image compatible with docker.
docker pull docker/whalesay	pull downloads whalesay image from docker hub I used pull command to download the k8s book image docker pull nigelpoulton/k8sbook
docker run - - name_you_select image_name cowsay Hello-world!	instantiate image into a container and execute it
docker run redis:4.0 Docker.hub.com find out all supported versions	Tag after semicolon specifies the version of the image you want to run as a container. If no tag specified docker will use as default tag :latest or the latest version
docker run -I *image_name*	The I (must be in lowercase) indicates container will run in interactive mode accepting input from stdin (like keyboard)
docker run -it *image_name*	Use it for interactive and display prompt if any
docker run -d *image_name* docker attach container_id	where d=detach or sbmjob like effect to reattach use attach command
docker run -p 80:5000 *image_name*	Mapping port #’s so users can access say a web app listening internally at port:5000 and internal IP address for user access http://ip_docker_host:80 and using the run cmd as shown to map port 5000 to 80

docker run -v /opt/datadir : /var/lib/mysql/ mysql (which is the image name) in this case the ext dir is /opt/datadir	Persist Data (so it will be there after container stopped) Must MAP external directory to internal mysql container directory
docker stop *container_name*	stops active container
docker rm container_name or id	removes container permanently
docker rmi *image name*	remove image example: rmi ubuntu
docker exec	run a cmd within a container

Environment Variables	
docker run -e APP_COLOR=blue *image_name*	-e exports the env_var named APP_COLOR and its bvlue is set to blue
docker inspect *container_name*	to display the env_vars in a running container - look for the Env: stanza or tag
docker containerize apps	
to build the image use cmd docker build	docker build *docker_file_name* -t /image_name
To publish itb to docker hub	push *image_name*
Docker File Format FROM (base-os-for-image) RUN (get all dependencies) COPY copies source code from local directory to the image ENTRYPOINT (first cmd to run when container is started)	Dockerfile FROM Ubuntu RUN apt-get update RUN apt-get install python RUN pip install flask RUN pip instal flask-mysql COPY . /opt/source-code ENTRYPOINT FLASK_APP=/opt/source-code/app.py flask run
docker history *image_name*	to view history of above Dockerfile combined commands

What can be containerized?	Not only OS like Ubuntu or DB’s like mysql or python but also browsers like Chrome or Firefox, apps like Skype, etc.
Find out contents of Dockerfile in this case search for the FROM command contains the base os or image to use	grep -I(lowercase) FROM | /root/webapp-color/Dockerfile cat will work too
Since OS & DB images will exit when using the Docker run cmd we can override it so they do not exit	docker run ubuntu sleep 5 normally cmd for ubuntu image is bsh which expects a terminal & when not found it exits. If we specify a cmd like sleep then the cmd will be executed prior to exiting ubuntu
To make the above permanent in the docker file used to build image	specify: CMD [“sleep”, “5"] (in JSON format)
To avoid having to specify the command & just supply the parameter “5" docker run ubuntu-sleeper 5	In the Dockerfile specify: FROM Ubuntu ENTRYPOINT [“sleep”]
To supply a default if entry not provided specify in Dokerfile	ENTRYPOINT [“sleep”] CMD [“5"]
docker network	
bridge ne a private **internal net created by docker in host where installed 172.17.n.n for each running container**	docker run Ubuntu - -network=none docker run Ubuntu - -network=host to access map the internal ports to the host using the -p parameter **default net = Bridge the other 2 host & none**
	another way is to associate the container with the (docker-host) ip so it can be accessed from outside world. BUT you will need to specify different ports since all containers will share same IP docker run Ubuntu - -network=host
	using the none network will isolate the container. No other container will be able to access it nor the outside world.

Creating multiple internal IP nets	docker network create driver - - bridge - - subnet 182.18.0.0/16 custom-isolated-network
docker network ls	display all nets
docker inspect *container_name* Networks: stanza	will display the IP settings of the net where container is running
Connect to another container using DNS Ex. A web container needs access to mysql db	mysql.connect (mysql) where mysql is the container name running mysql
Storage Access	
File System docker	/var/lib/docker/ /aufs /containers /image /volumes
Persistence (Volumes)	docker volume create data_volume will be created in /var/lib/docker/volumes/data_volume
Mount Volume when running the container Mount directory or complete path to folder we want to save data to	docker run -v data_volume:/var/lib/mysql mysql (this is known as volume mount) docker run -v /data/mysql : /var/lib/mysql mysql will write data to dir /data/mysql (bind mount)
	a more modern approach to the -v flag is to use the mount keyword like this: docker run - -mount type=bind ,source=/data/mysql, target=/var/lib/mysql mydsql

Labs: kodekloud.com/p/docker-labs Kloudjvh0412

Note: to shutdown docker go to the tray located at the bottom-right of Windows Desktop, right click the docker icon and select Quit Docker. If you want to see the docker settings GUI panel use same icom and select settings.

Note: when you are in a given context, (like your current directory), commands will be sent to the cluster associated with that context. You will be posing commands (YAML) to the API Server associated with the cluster defined in the context.

GitHub

GitHub is a Git repository hosting service, but it adds many of its own features. While Git is a command line tool, GitHub provides a Web-based graphical interface. It also provides access control and several collaboration features, such as a wikis and basic task management tools for every project.

When you put code on GitHub, you retain all the copyright to your code. However, you do grant GitHub a license to host the code, and you also allow GitHub users a set of rights - namely the ability to look at, and fork your repository. ... This gives others rights to your code, but you still retain the copyright

GitHub is an online repository service that anyone can sign up for. ... For open source projects, GitHub is entirely free to use, which is great for programmers and collaborators. It's rather easy to look at and edit code on GitHub, and it's widespread use by coders has made it one of the biggest open source communities

Microsoft acquired GitHub, a popular code-repository service used by many developers and large companies, for $7.5 billion in stock.

GitHub is the best place to share code with friends, co-workers, classmates, and complete strangers. Over three million people use GitHub to build amazing things together. GitHub is a tool in the Code Collaboration & Version Control category of a tech stack.

Git es un software de control de versiones diseñado por Linus Torvalds, pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando éstas tienen un gran número de archivos de código fuente.