



Docker Introduction

Getting Started With Docker

Topics

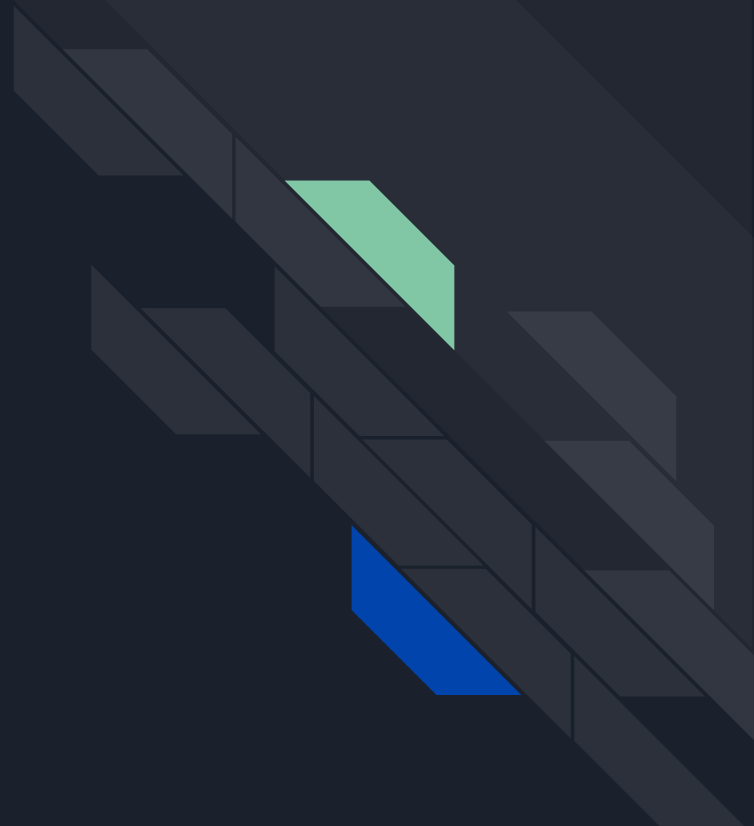
What is Docker?

Docker Engine

Concepts

Testing Docker

Docker and Python





What Is Docker?



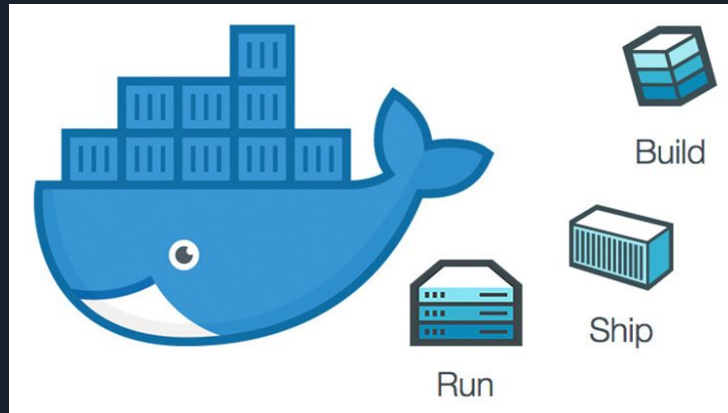
What is Docker

Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly. With Docker, you can manage your infrastructure in the same ways you manage your applications. By taking advantage of Docker's methodologies for shipping, testing, and deploying code quickly, you can significantly reduce the delay between writing code and running it in production.



What is Docker

Basically, Docker is a tool that allows developers, sys-admins etc. to easily deploy their applications in a sandbox (called containers) to run on the host operating system i.e. Linux. Docker is a platform for developers and sysadmins to build, run, and share applications with containers. The use of containers to deploy applications is called containerization. Containers are not new, but their use for easily deploying applications is.

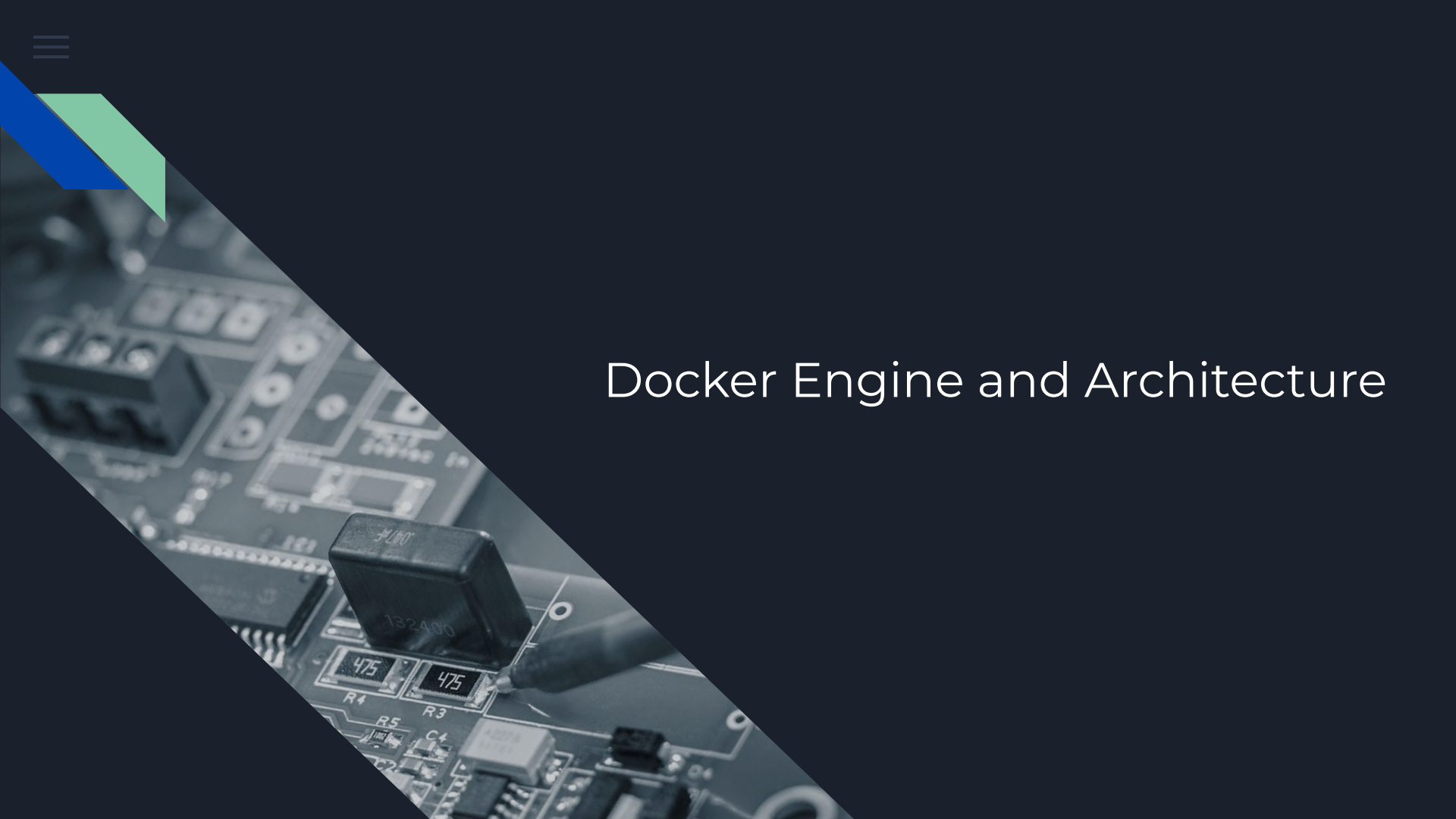




What is Docker

Containerization is increasingly popular because containers are:

- **Flexible:** Even the most complex applications can be containerized.
- **Lightweight:** Containers leverage and share the host kernel, making them much more efficient in terms of system resources than virtual machines.
- **Portable:** You can build locally, deploy to the cloud, and run anywhere.
- **Loosely coupled:** Containers are highly self sufficient and encapsulated, allowing you to replace or upgrade one without disrupting others.
- **Scalable:** You can increase and automatically distribute container replicas across a datacenter.
- **Secure:** Containers apply aggressive constraints and isolations to processes without any configuration required on the part of the user.



Docker Engine and Architecture

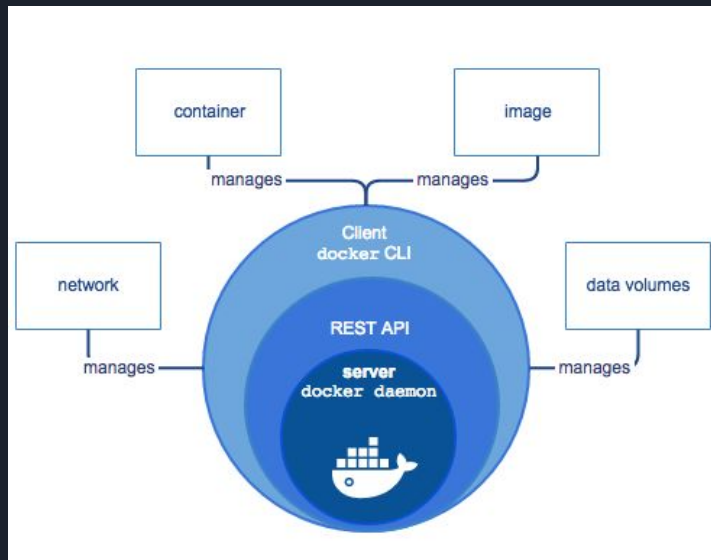


Docker Engine

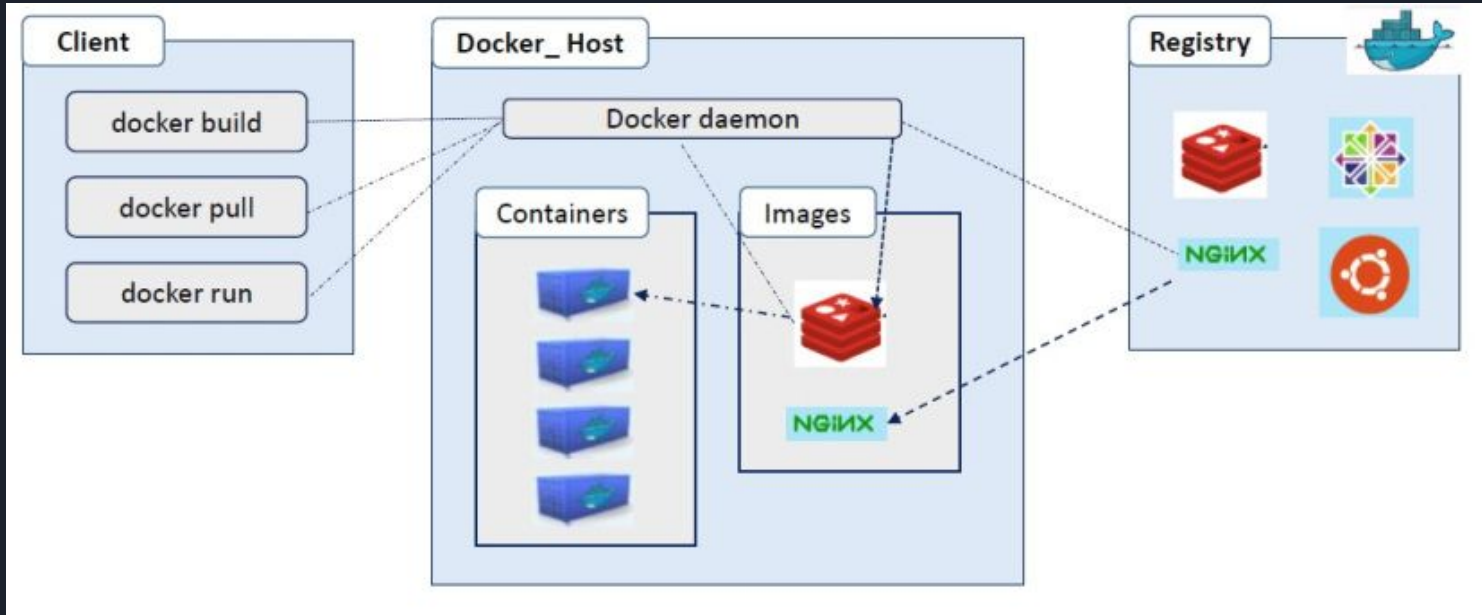
Docker Engine is a client-server application with these major components:

- A server which is a type of long-running program called a daemon process (the `dockerd` command).
- A REST API which specifies interfaces that programs can use to talk to the daemon and instruct it what to do.
- A command line interface (CLI) client (the `docker` command).

Docker Engine



Docker Architecture

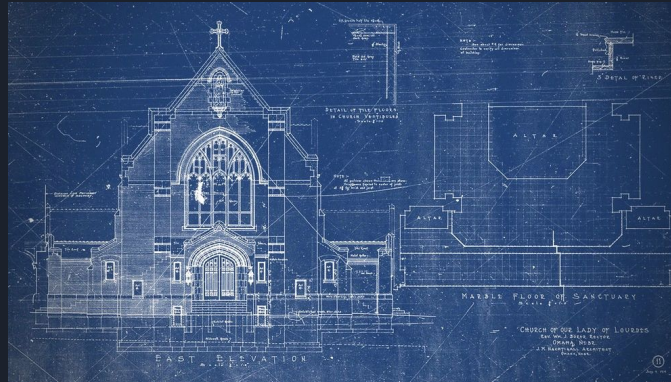




Docker Concepts

Docker Image

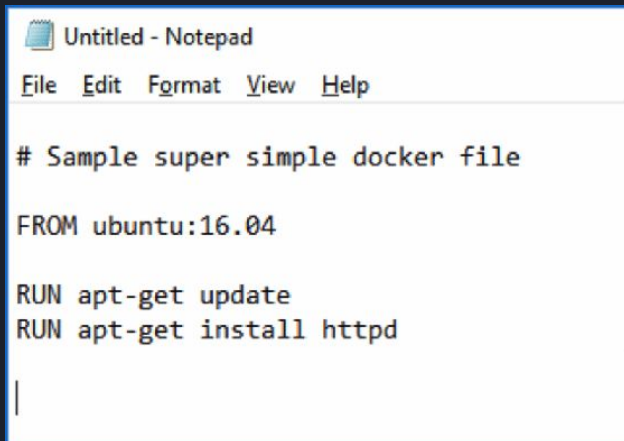
An image is a static representation of the app or service and its configuration and dependencies. A Docker image is built up from a series of layers, allowing a minimal amount of data to be sent when transferring images over network.





Dockerfile

A Dockerfile is a text document that contains all the commands a user could call on the command line to assemble an image.



```
Untitled - Notepad
File Edit Format View Help

# Sample super simple docker file

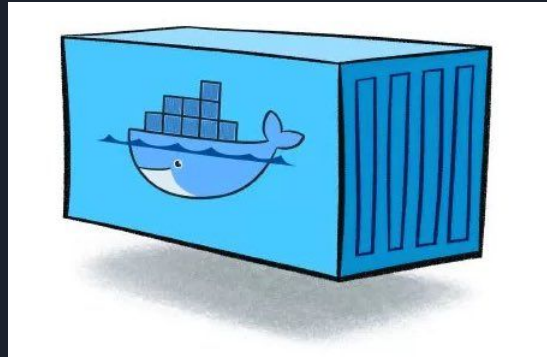
FROM ubuntu:16.04

RUN apt-get update
RUN apt-get install httpd

|
```

Docker Container

Container is runtime object or representation of an image. Containers are lightweight & portable encapsulation of an environment where applications are run.

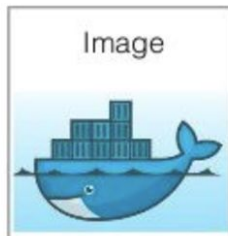


Docker Pipeline

```
FROM ubuntu:14.04
MAINTAINER John Doe <john.doe@example.com>
RUN echo "Hello, World!" > /usr/local/bin/hello
RUN echo "Hello, World!" > /usr/local/bin/hello
RUN echo "Hello, World!" > /usr/local/bin/hello
RUN echo "Hello, World!" > /usr/local/bin/hello
RUN echo "Hello, World!" > /usr/local/bin/hello
RUN echo "Hello, World!" > /usr/local/bin/hello
RUN echo "Hello, World!" > /usr/local/bin/hello
RUN echo "Hello, World!" > /usr/local/bin/hello
RUN echo "Hello, World!" > /usr/local/bin/hello
RUN echo "Hello, World!" > /usr/local/bin/hello
```

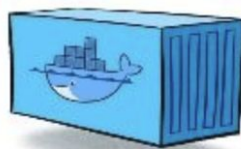
Dockerfile

build



Docker Image

run



Docker Container



Testing Docker



Installation

Windows 10 Professional:

- Instalacion: <https://www.youtube.com/watch?v=BK-C2RofmTE>
- documentacion: <https://docs.docker.com/docker-for-windows/>

Windows 10 Home Edition:

- Instalacion: <https://www.youtube.com/watch?v=YH3sutAsxEM>
- Documentacion: <https://docs.docker.com/docker-for-windows/install-windows-home/>

MacOs:

- Instalacion: https://www.youtube.com/watch?v=mbSsh40_8WM
- Documentacion: <https://docs.docker.com/docker-for-mac/install/>



Your First Container

```
$ docker run hello-world
```

```
Hello from Docker.
```

```
This message shows that your installation appears to be working correctly.
```

```
...
```



Your Second Container

```
$ docker pull busybox
```



Checking Images

```
$ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	VIRTUAL
busybox	latest	c51f86c28340	4 weeks ago	1.109 MB



Running Your Second Container

```
$ docker run busybox  
$
```



Running With Extra Parameters

```
$ docker run busybox echo "hello from busybox"  
hello from busybox
```



Check Running Containers

```
$ docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS
--------------	-------	---------	---------	--------



Check stopped containers

```
$ docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS
305297d7a235	busybox	"uptime"	11 minutes ago	Exited (0)
ff0a5c3750b9	busybox	"sh"	12 minutes ago	Exited (0)
14e5bd11d164	hello-world	"/hello"	2 minutes ago	Exited (0)



Run and get inside the container

```
$ docker run -it busybox sh
/ # ls
bin  dev  etc  home  proc  root  sys  tmp  usr  var
/ # uptime
05:45:21 up 5:58, 0 users, load average: 0.00, 0.01, 0.04
```



Delete Containers

```
$ docker rm 305297d7a235 ff0a5c3750b9  
305297d7a235  
ff0a5c3750b9
```



Delete All

```
$ docker container prune
WARNING! This will remove all stopped containers.
Are you sure you want to continue? [y/N] y
Deleted Containers:
4a7f7eebae0f63178aff7eb0aa39f0627a203ab2df258c1a00b456cf20063
f98f9c2aa1eaf727e4ec9c0283bcaa4762fbdba7f26191f26c97f64090360

Total reclaimed space: 212 B
```



Docker and Python

Install Jupyter

```
obed@OESPINOZA: ~  
File Edit View Search Terminal Help  
obed@OESPINOZA:~$ docker pull jupyter/base-notebook  
Using default tag: latest  
latest: Pulling from jupyter/base-notebook  
da7391352a9b: Download complete  
14428a6d4bcd: Download complete  
2c2d948710f2: Download complete  
e3cbfeece0ae: Downloading 1.231MB/10.23MB  
48bd2a353bd8: Download complete  
235d93b8ccf1: Waiting  
4f4fb700ef54: Waiting  
b6c06056c45b: Waiting  
60918bcbe6d4: Waiting  
762f9ebe4ddc: Waiting  
1df9d491a039: Waiting  
be84c8c720e3: Waiting  
28807e96859d: Waiting  
bcdaf848f29a: Waiting  
49777cff52f1: Waiting  
7fb3bffa2e73: Waiting  
█
```



Run Jupyter

- `docker run -p 8888:8888 jupyter/base-notebook`

← → ↻ ⓘ 127.0.0.1:8888/tree



 jupyter


Quit

Logout

Files **Running** Clusters

Select items to perform actions on them.

Upload New ↕

☐ 0 ▾  /

Name ▾

Last Modified

File size

☐  work

7 days ago



Create a Network

- `docker network create --driver bridge my_test_network`

```
obed@0ESPINOZA:~$ docker network create --driver bridge my_test_network
f4afb7fa41d1f41737f516b55b4e9e62908df1513e4974ec639f7593cd176703
obed@0ESPINOZA:~$
```



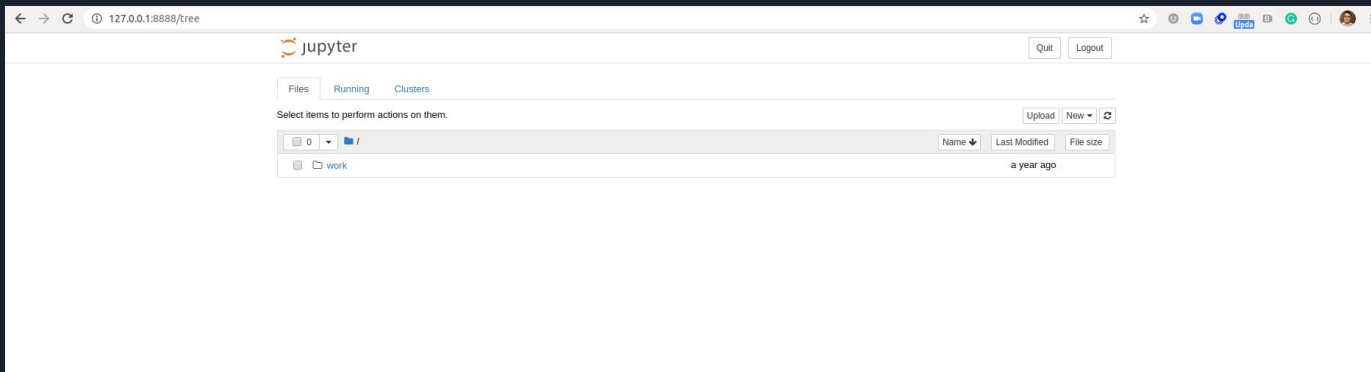
Run Mysql Container

- `docker run -it --network my_test_network -e "MYSQL_ROOT_PASSWORD=root123" -e "MYSQL_DATABASE=test" -e "MYSQL_USER=test" -e "MYSQL_PASSWORD=test123" mysql:5.7.35`

```
obed@OESPINOZA:~$ docker run -it --network my_test_network -e "MYSQL_ROOT_PASSWORD=root123" -e "MYSQL_DATABASE=test" -e "MYSQL_USER=test" -e "MYSQL_PASSWORD=test123" mysql
2020-08-04 02:08:39+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.21-1debian10 started.
2020-08-04 02:08:39+00:00 [Note] [Entrypoint]: Switching to dedicated user 'mysql'
2020-08-04 02:08:39+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.21-1debian10 started.
2020-08-04 02:08:39+00:00 [Note] [Entrypoint]: Initializing database files
2020-08-04T02:08:39.822957Z 0 [System] [MY-013169] [Server] /usr/sbin/mysqld (mysqld 8.0.21) initializing of server in progress as process 43
2020-08-04T02:08:39.828862Z 1 [System] [MY-013576] [InnoDB] InnoDB initialization has started.
2020-08-04T02:08:41.491304Z 1 [System] [MY-013577] [InnoDB] InnoDB initialization has ended.
2020-08-04T02:08:45.050870Z 6 [Warning] [MY-010453] [Server] root@localhost is created with an empty password ! Please consider switching off the --initialize-insecure option.
2020-08-04 02:08:53+00:00 [Note] [Entrypoint]: Database files initialized
2020-08-04 02:08:53+00:00 [Note] [Entrypoint]: Starting temporary server
```


Run Jupyter With Network

- `docker run --network my_test_network -p 8888:8888 jupyter/base-notebook`





Check Running Containers

- Docker ps

```
obed@0ESPINOZA:~$ docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
bac8401dab67	jupyter/scipy-notebook:17aba6048f44	"tini -g -- start-no..."	28 seconds ago	Up 26 seconds	0.0.0.0:8888->8888/tcp	laughing_borg
68acdb620f0c	mysql	"docker-entrypoint.s..."	9 minutes ago	Up 9 minutes	3306/tcp, 33060/tcp	affectionate_johnson



Accessing Jupyter Container

- `docker exec -it laughing_borg sh`

```
obed@OESPINOZA:~$ docker exec -it laughing_borg sh
$
```



Install Mysql Connector

- `pip install mysql-connector-python`

```
$ pip install mysql-connector-python
Collecting mysql-connector-python
  Downloading https://files.pythonhosted.org/packages/ce/1d/378f92f45fb98d7e033f6a94b0600b8ae496e67ba411b992f536784a3d0d/mysql_connector_python-8.0.21-cp36-cp36m-manylinux1_x86_64.whl (15.8MB)
    100% |#####| 15.8MB 2.4MB/s
Requirement already satisfied: protobuf>=3.0.0 in /opt/conda/lib/python3.6/site-packages (from mysql-connector-python) (3.6.1)
Requirement already satisfied: six>=1.9 in /opt/conda/lib/python3.6/site-packages (from protobuf>=3.0.0->mysql-connector-python) (1.12.0)
Requirement already satisfied: setuptools in /opt/conda/lib/python3.6/site-packages (from protobuf>=3.0.0->mysql-connector-python) (40.6.3)
Installing collected packages: mysql-connector-python
Successfully installed mysql-connector-python-8.0.21
$
```

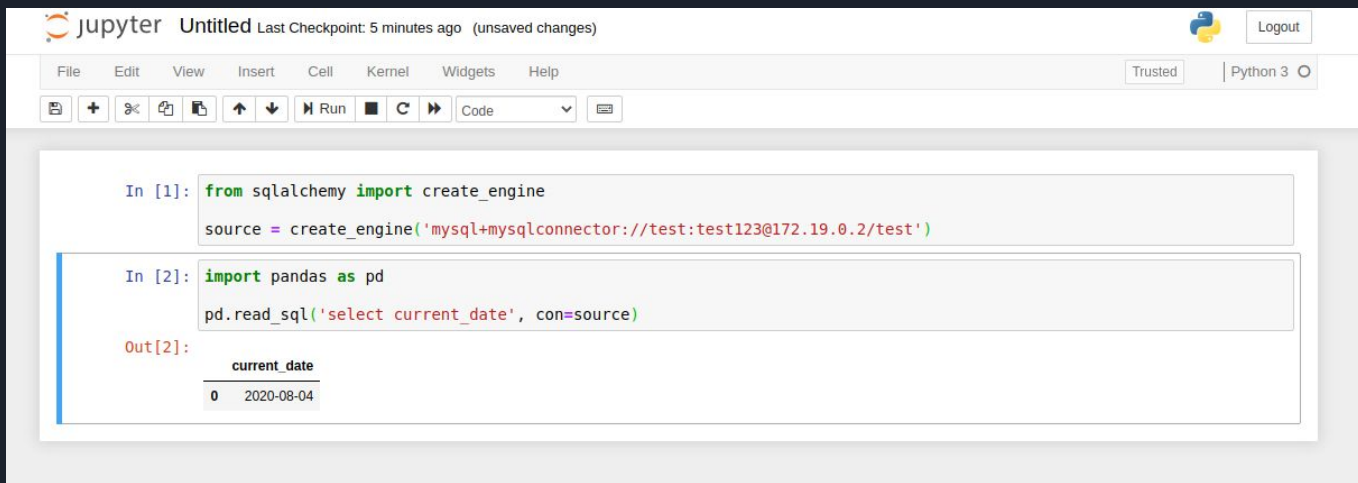


Check Network

- `docker network inspect my_test_network`

```
"Internal": false,
"Attachable": false,
"ConfigFrom": {
  "Network": ""
},
"ConfigOnly": false,
"Containers": {
  "68acdb620f0c158837574a75613733ed7b922dae3cbc5766e5dd4bb7e0f3b80a": {
    "Name": "affectionate_johnson",
    "EndpointID": "e7b8d73fd433d14877013d5b9b467a0a5d48bbffa9fef79a29855080b43c985c",
    "MacAddress": "02:42:ac:13:00:02",
    "IPv4Address": "172.19.0.2/16",
    "IPv6Address": ""
  },
  "bac8401dab67f16662d9d2e031096a9b86ba821110f1ae1a094f46c524262da9": {
    "Name": "laughing_borg",
    "EndpointID": "5a8b077397af31ac4386da4b09d7b4bfe2b4465dd6ceb4a77b123f30c6124624",
    "MacAddress": "02:42:ac:13:00:03",
    "IPv4Address": "172.19.0.3/16",
    "IPv6Address": ""
  }
},
"Options": {},
"Labels": {}
}
```

Test our connection



The image shows a Jupyter Notebook interface with the title "Untitled" and a status bar indicating "Last Checkpoint: 5 minutes ago (unsaved changes)". The interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help) and a toolbar with icons for file operations, running, and code execution. The notebook contains two input cells and one output cell.

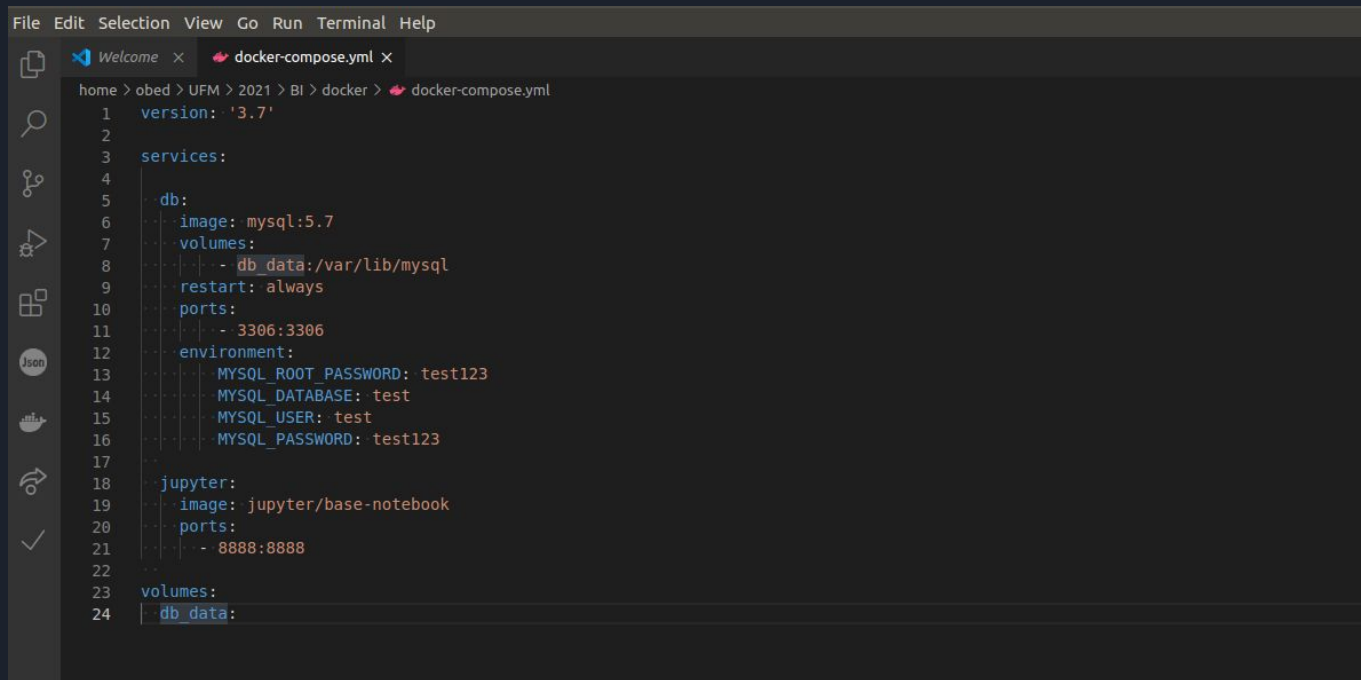
```
In [1]: from sqlalchemy import create_engine
        source = create_engine('mysql+mysqlconnector://test:test123@172.19.0.2/test')
```

```
In [2]: import pandas as pd
        pd.read_sql('select current_date', con=source)
```

Out[2]:

	current_date
0	2020-08-04

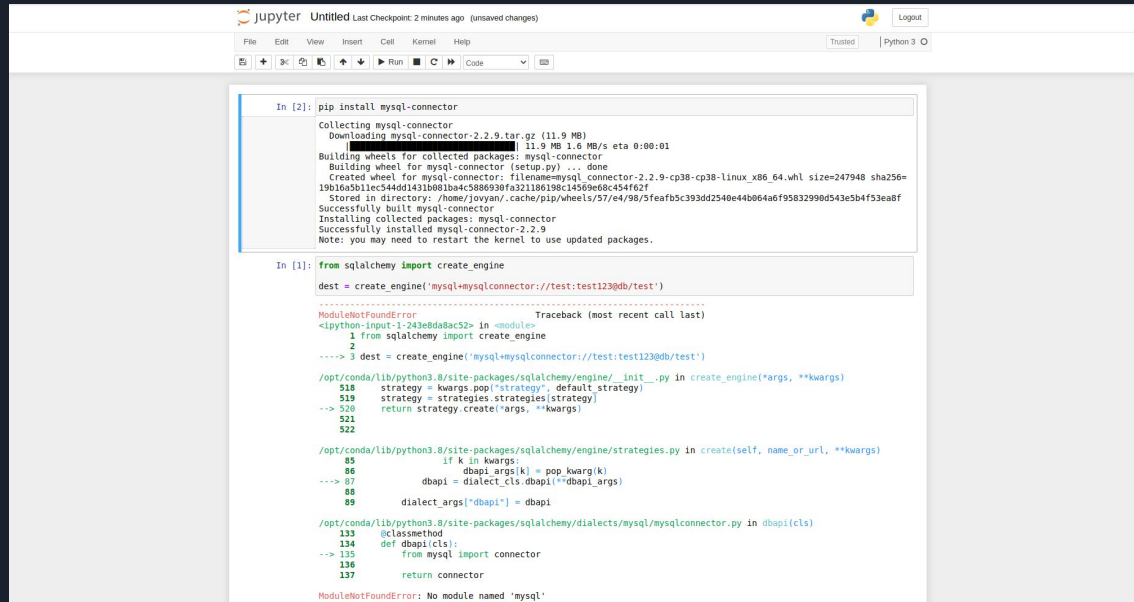
Introducing Docker Compose



The image shows a code editor window with a dark theme. The menu bar at the top includes File, Edit, Selection, View, Go, Run, Terminal, and Help. The tab bar shows two tabs: 'Welcome' and 'docker-compose.yml'. The editor displays the content of 'docker-compose.yml' with line numbers 1 through 24. The file defines two services: 'db' (MySQL) and 'jupyter'. The 'db' service uses the 'mysql:5.7' image, maps port 3306 to 3306, and sets environment variables for root password, database name, and user. The 'jupyter' service uses the 'jupyter/base-notebook' image and maps port 8888 to 8888. A shared volume 'db_data' is defined at the bottom.

```
1  version: '3.7'
2
3  services:
4
5    db:
6      image: mysql:5.7
7      volumes:
8        - db_data:/var/lib/mysql
9      restart: always
10     ports:
11       - 3306:3306
12     environment:
13       - MYSQL_ROOT_PASSWORD: test123
14       - MYSQL_DATABASE: test
15       - MYSQL_USER: test
16       - MYSQL_PASSWORD: test123
17
18     jupyter:
19       image: jupyter/base-notebook
20       ports:
21         - 8888:8888
22
23  volumes:
24    db_data:
```

Install Mysql Client



```
jupyter Untitled Last Checkpoint: 2 minutes ago (unsaved changes)
File Edit View Insert Cell Kernel Help Trusted Python 3
In [2]: pip install mysql-connector
Collecting mysql-connector
  Downloading mysql-connector-2.2.9.tar.gz (11.9 MB)
    11.9 MB 1.6 MB/s eta 0:00:01
Building wheels for collected packages: mysql-connector
  Building wheel for mysql-connector (setup.py) ... done
  Created wheel for mysql-connector: filename=mysql_connector-2.2.9-cp38-cp38-linux_x86_64.whl size=247948 sha256=
  19b16a5b1ec544dd1431b681b4c586938fa321186190c14509e68c434f62f
  Stored in directory: /home/jovyan/.cache/pip/wheels/57/e4/98/5feaf5bc393dd2540e44b064a6f95832996d543e5b4f53ea8f
Successfully built mysql-connector
Installing collected packages: mysql-connector
Successfully installed mysql-connector-2.2.9
Note: you may need to restart the kernel to use updated packages.

In [1]: from sqlalchemy import create_engine

dest = create_engine('mysql+mysqlconnector://test:test123@db/test')

.....
ModuleNotFoundError                               Traceback (most recent call last)
<ipython-input-1-243bda8ac52> in <module>
      1 from sqlalchemy import create_engine
      2
----> 3 dest = create_engine('mysql+mysqlconnector://test:test123@db/test')

/opt/conda/lib/python3.8/site-packages/sqlalchemy/engine/_init_.py in create_engine(*args, **kwargs)
   518 strategy = kwargs.pop("strategy", default_strategy)
   519 strategy = strategies.strategies[strategy]
--> 520 return strategy.create(*args, **kwargs)
   521
   522

/opt/conda/lib/python3.8/site-packages/sqlalchemy/engine/strategies.py in create(self, name_or_url, **kwargs)
    85 if k in kwargs:
    86     dbapi_args[k] = pop_kwarg(k)
--> 87     dbapi = dialect_cls.dbapi(**dbapi_args)
    88
    89     dialect_args["dbapi"] = dbapi

/opt/conda/lib/python3.8/site-packages/sqlalchemy/dialects/mysql/mysqlconnector.py in dbapi(cls)
   133 @classmethod
   134 def dbapi(cls):
--> 135     from mysql import connector
   136
   137     return connector

ModuleNotFoundError: No module named 'mysql'
```


Install Pandas

```
jupyter Untitled Last Checkpoint: 3 minutes ago (unsaved changes)
File Edit View Insert Cell Kernel Help Trusted Python 3
In [2]: pip install mysql-connector
Collecting mysql-connector
  Downloading mysql-connector-2.2.9.tar.gz (11.9 MB)
    [REDACTED] 11.9 MB 1.6 MB/s eta 0:00:01
Building wheels for collected packages: mysql-connector
  Building wheel for mysql-connector (setup.py) ... done
  Created wheel for mysql-connector: filename=mysql_connector-2.2.9-cp38-cp38-linux_x86_64.whl size=247948 sha256=
  19016a501ec544d014318081b4dc5806930fa21186190c14566e6dc454fe2f
  Stored in directory: /home/jovyan/.cache/pip/wheels/57/e4/98/5feafb5c393dd2540e44b664a6f95832990d543e5b4f53ea8f
Successfully built mysql-connector
Installing collected packages: mysql-connector
Successfully installed mysql-connector-2.2.9
Note: you may need to restart the kernel to use updated packages.

In [3]: pip install pandas
Collecting pandas
  Downloading pandas-1.2.1-cp38-cp38-manylinux1_x86_64.whl (9.7 MB)
    [REDACTED] 9.7 MB 873 kB/s eta 0:00:01
Requirement already satisfied: python-dateutil<=2.7.3 in /opt/conda/lib/python3.8/site-packages (from pandas) (2.
8.1)
Collecting numpy>=1.16.5
  Downloading numpy-1.19.5-cp38-cp38-manylinux2010_x86_64.whl (14.9 MB)
    [REDACTED] 14.9 MB 9.5 MB/s eta 0:00:01
Requirement already satisfied: six>=1.5 in /opt/conda/lib/python3.8/site-packages (from python-dateutil<=2.7.3->pa
ndas) (1.15.0)
Collecting pytz>=2017.3
  Downloading pytz-2020.5-py2.py3-none-any.whl (510 kB)
    [REDACTED] 510 kB 5.1 MB/s eta 0:00:01
Installing collected packages: pytz, numpy, pandas
Successfully installed numpy-1.19.5 pandas-1.2.1 pytz-2020.5
Note: you may need to restart the kernel to use updated packages.

In [1]: from sqlalchemy import create_engine
dest = create_engine('mysql+mysqlconnector://test:test123@db/test')

In [2]: import pandas as pd
pd.read_sql('select current_date', con=dest)

ModuleNotFoundError Traceback (most recent call last)
<ipython-input-2-b39996035d15> in <module>
----> 1 import pandas as pd
      2
      3 pd.read_sql('select current_date', con=dest)
```