



Selenium Grid with Docker



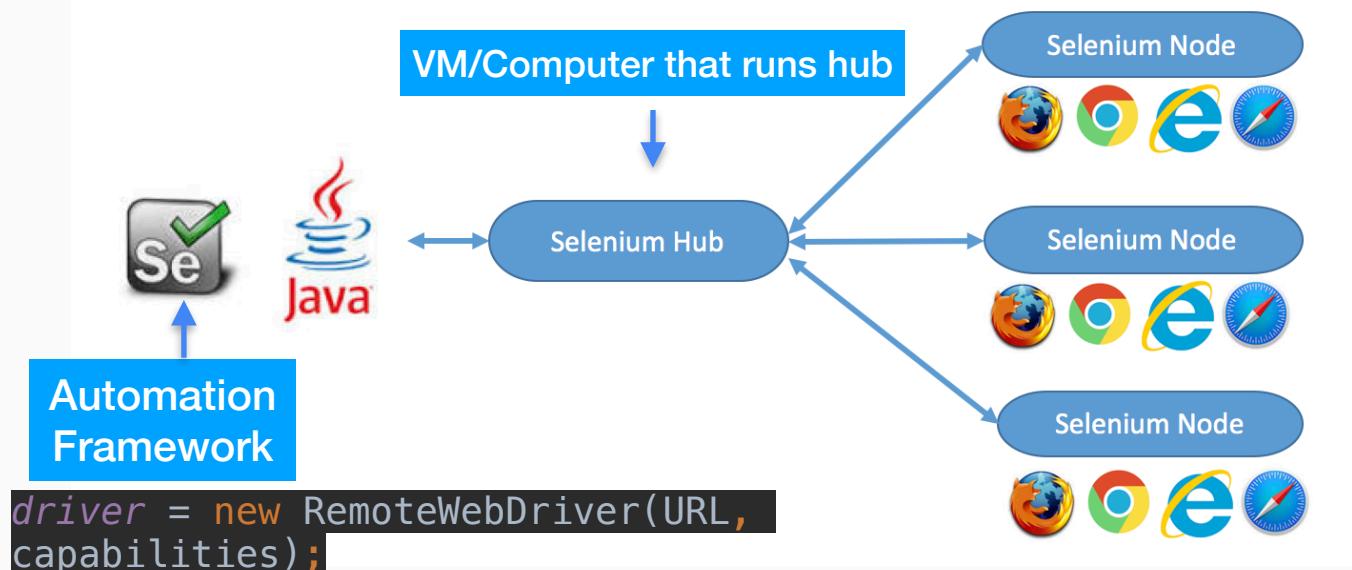
CYDEO

SELENIUM GRID



What is Selenium Grid ?

The **Selenium Grid** is a testing tool which allows us to run our tests on different machines against different browsers. It is a part of the Selenium Suite which specialize in running multiple tests across different browsers, operating system and machines. You can connect to it with Selenium Remote by specifying the browser, browser version, and operating system you want. You specify these values through Selenium Remote's Capabilities. There are two main elements to **Selenium Grid** – a **hub** and **node**.



Every node runs on different VM/Server. Connection happens based on IP address of the hub.

Node can be running on same machine as hub.



What is Selenium Grid ?

Grid allows you to :

- scale by distributing tests on several machines (parallel execution)
- manage multiple environments from a central point, making it easy to run the tests against a vast combination of browsers / OS.
- minimize the maintenance time for the grid by allowing you to implement custom hooks to leverage virtual infrastructure for instance.



HUB

- The hub is the central point where you load your tests into.
- There should only be one hub in a grid.
- The hub is launched only on a single machine, say, a computer whose O.S is Windows 10 and whose browser is Chrome.
- The machine containing the hub is where the tests will be run, but you will see the browser being automated on the node.
- The hub can be parametrized with json file.



NODE

- Nodes are the Selenium instances that will execute the tests that you loaded on the hub.
- There can be one or more nodes in a grid.
- Nodes can be launched on multiple machines with different platforms and browsers.
- Nodes can be located on different machines.
- Nodes can be parametrized with json file.



How Selenium Grid Works

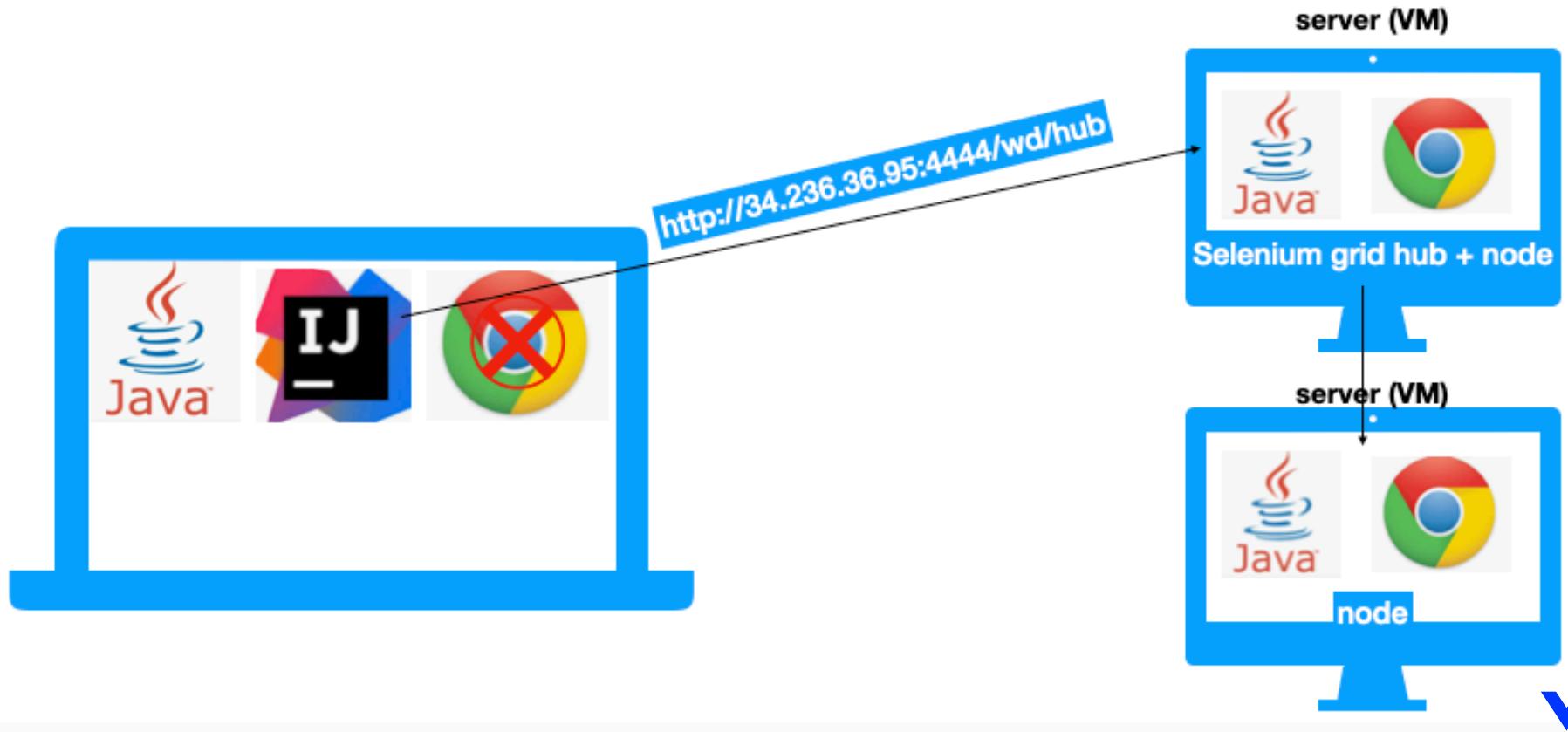
A grid consists of a single hub, and one or more nodes. Both are started using the **selenium-server.jar executable**.

The hub receives a test to be executed along with information on which browser and ‘platform’ (i.e. WINDOWS, LINUX, etc) where the test should be run. It ‘knows’ the configuration of each node that has been ‘registered’ to the hub. Using this information it selects an available node that has the requested browser-platform combination.

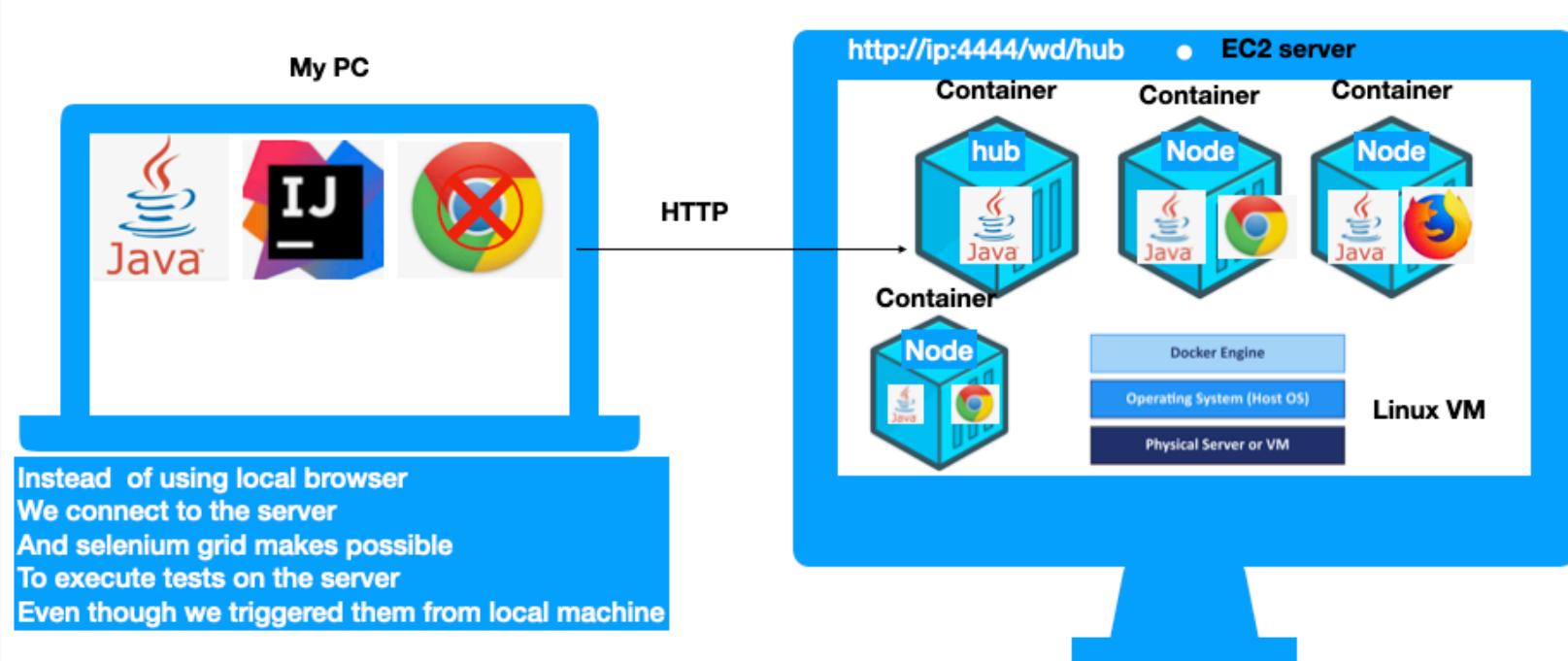
Once a node has been selected, Selenium commands initiated by the test are sent to the hub, which passes them to the node assigned to that test. The node runs the browser, and executes the Selenium commands within that browser against the application under test.



Execution Happens on the Server



Execution Happens on the Server



Configure the nodes by Command Line

By default, starting the node allows for concurrent use of 11 browsers... : 5 Firefox, 5 Chrome, 1 Internet Explorer. The maximum number of concurrent tests is set to 5 by default.

To change this and other browser settings, you can pass in parameters to each -browser switch (each switch represents a node based on your parameters). If you use the -browser parameter, the default browsers will be ignored and only what you specify command line will be used.

-browser

browserName=firefox,version=3.6,maxInstances=5,platform=LINUX



Terminologies used in Grid Configuration

role = When launching a node, it will forward the parameter to server on the node like, -role node.

host = Though it not needed and determined automatically. For some network configuration, network with VPN, specifying the host might be necessary.

port = the port the remote/hub will listen on. Default to 4444.

throwIn Capability Not Present = Default value is true; if no proxy is currently registered hub will reject the test request. On contrast, the request queued until a node supporting the capability which is added to the grid.

new Session Wait_Timeout = Default to no timeout (-1)ms after which a new test waiting for a node to become available will timeout else test will throw an exception before starting a browser.

hubConfig = It defines the hub properties in JSON format.

nodeConfig = It defines the node properties in JSON format.

cleanupCycle = It will check the timeout thread in ms.

node Timeout = The timeout in seconds prior to hub automatically ends the test so that browser will be released for another test to run.

browser Timeout= The browser gets hang after defined timeout in ms.

hub = It used to post the registration request using url – <<http://localhost:4444/grid/register>>

proxy =This will be used to represent the node. By default org.openqa.grid.selenium.proxy.DefaultRemoteProxy.

maxSession = Maximum number of sessions to run multiple tests at same time independently in different browsers.

register Cycle = It defines how often registered node will try to register itself again in ms,however, without restarting the node command file.

nodePolling = How often the hub checks if the node is still alive in ms.



CYDEO

AWS



What is AWS ?

Amazon Web Services (AWS) is a secure cloud services platform, offering compute power, database storage, content delivery and other functionality to help businesses scale and grow.

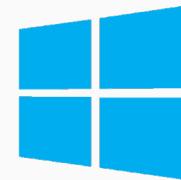
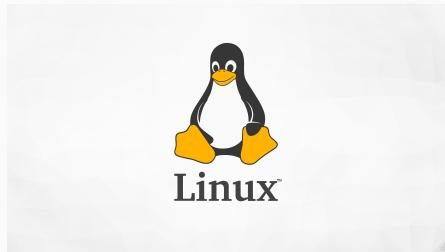
In simple words AWS allows you to do the following things

- Running web and application servers in the cloud to host dynamic websites.
- Securely store all your files on the cloud so you can access them from anywhere.
- Using managed databases like MySQL, PostgreSQL, Oracle or SQL Server to store information.
- Deliver static and dynamic files quickly around the world using a Content Delivery Network (CDN).
- Send bulk email to your customers



What is AWS ?

Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides secure, resizable compute capacity in the cloud. It is designed to make web-scale cloud computing easier for developers. Amazon EC2's simple web service interface allows you to obtain and configure capacity with minimal friction. It provides you with complete control of your computing resources and lets you run on Amazon's proven computing environment.



Windows
Server



What is Linux ?

From smartphones to cars, supercomputers and home appliances, home desktops to enterprise servers, the Linux operating system is everywhere.

Linux has been around since the mid-1990s and has since reached a user-base that spans the globe. Linux is actually everywhere: It's in your phones, your thermostats, in your cars, refrigerators, Roku devices, and televisions. It also runs most of the Internet, all of the world's top 500 supercomputers, and the world's stock exchanges.

But besides being the platform of choice to run desktops, servers, and embedded systems across the globe, Linux is one of the most reliable, secure and worry-free operating systems available.

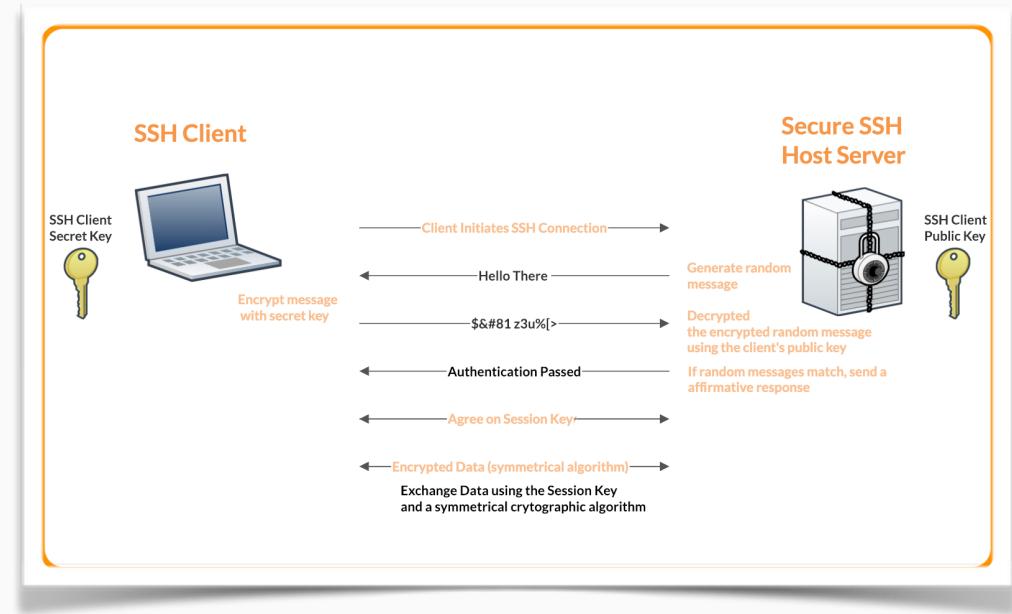
Just like Windows, iOS, and Mac OS, Linux is an operating system. In fact, one of the most popular platforms on the planet, Android, is powered by the Linux operating system. An operating system is software that manages all of the hardware resources associated with your desktop or laptop. To put it simply, the operating system manages the communication between your software and your hardware. Without the operating system (OS), the software wouldn't function.



SSH (Secure Shell)

An SSH client allows you to connect to a remote computer running an SSH server.

The Secure Shell (SSH) protocol is often used for remote terminal connections, allowing you to access a text-mode terminal on a remote computer as if you were sitting of it.





Launch EC2 INSTANCE (GRID)



Launch EC2

The screenshot shows the AWS console interface. A red arrow points from the search bar at the top left to the search results section. The search bar contains the text 'EC2'. The results are filtered under the 'Services' category, showing two items: 'EC2' and 'EC2 Image Builder'. The 'EC2' item is highlighted with a blue border.

aws Services Search EC2 X

Console Actions ▾

Services (9)

Features (40)

Blogs (1,734)

Documentation (126,461)

Knowledge Articles (30)

Tutorials (18)

Events (0)

Search results for 'EC2'

Services See all 9 results ▾

EC2 ☆ Virtual Servers in the Cloud

EC2 Image Builder ☆ A managed service to automate build, customize and deploy OS images

- Type EC2 in search bar
- Choose EC2 from list



Launch EC2

The screenshot shows the AWS EC2 Dashboard. On the left, there's a sidebar with various navigation links like EC2 Dashboard, Instances, and Images. The main area displays metrics for running instances, elastic IPs, key pairs, placement groups, snapshots, dedicated hosts, instances, load balancers, security groups, and volumes. A callout box provides information about launching Microsoft SQL Server Always On availability groups. Below these metrics, there's a 'Launch instance' section with a large orange 'Launch Instance' button, which has a red arrow pointing to it from the bottom-left. There's also a 'Migrate a server' button. To the right, there's a 'Service health' section showing the status as 'This service is operating normally'.

- Click Launch Instance



Launch EC2

Name and tags

Name

 [Add additional tags](#)

- Type here your server name (name is not an issue)



Launch EC2

▼ Application and OS Images (Amazon Machine Image) [Info](#)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Search our full catalog including 1000s of application and OS images

Recent

Quick Start

Amazon Linux

Ubuntu

Windows

Red Hat

SUSE Linux

aws

ubuntu®

Microsoft

Red Hat

SUSE

Browse more AMIs

Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Amazon Linux 2 AMI (HVM) - Kernel 5.10, SSD Volume Type

ami-0022f774911c1d690 (64-bit (x86)) / ami-0e449176cecc3e577 (64-bit (Arm))

Virtualization: hvm ENA enabled: true Root device type: ebs

Free tier eligible

Description

Amazon Linux 2 Kernel 5.10 AMI 2.0.20220426.0 x86_64 HVM gp2

Architecture

64-bit (x86)

AMI ID

ami-0022f774911c1d690

- Choose Default One

Amazon Linux Server



▼ Instance type Info

Instance type

t2.micro

Family: t2 1 vCPU 1 GiB Memory

On-Demand Linux pricing: 0.0116 USD per Hour

On-Demand Windows pricing: 0.0162 USD per Hour

Free tier eligible

Compare instance types

NOTES

- Make sure that Instance type is chosen as **t2.micro free tier eligible**
- Only one type of server is free
750 hours per month for 12 month -> only for first year
 $31 \text{ day} \times 24 \text{ hours} = 744$
It means you need to have **only one server at a time**

 **Free tier:** In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 30 GiB of EBS storage, 2 million IOs, 1 GB of snapshots, and 100 GB of bandwidth to the internet 



Launch EC2

▼ Key pair (login) [Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - *required*

Select ▼

 [Create new key pair](#)

For now just skip this step. We will handle this after click **Launch instance**



Launch EC2

▼ Network settings

Network

vpc-34ff5649

Subnet

No preference (Default subnet in any availability zone)

Auto-assign public IP

Enable

Security groups (Firewall) [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

We'll create a new security group called '**launch-wizard-1**' with the following rules:

Allow SSH traffic from

Helps you connect to your instance

Anywhere

0.0.0.0/0

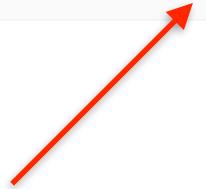
Allow HTTPS traffic from the internet

To set up an endpoint, for example when creating a web server

Allow HTTP traffic from the internet

To set up an endpoint, for example when creating a web server

Edit



- Click Edit button

⚠ Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only. **X**



Launch EC2

Inbound security groups rules

▼ Security group rule 1 (TCP, 22, 0.0.0.0/0) Remove

Type <small>Info</small>	Protocol <small>Info</small>	Port range <small>Info</small>
ssh	TCP	22
Source type <small>Info</small>	Source <small>Info</small>	Description - optional <small>Info</small>
Anywhere	<input style="width: 150px; height: 25px; border: 1px solid #ccc; border-radius: 5px; padding: 5px;" type="text" value="Add CIDR, prefix list or security"/> 0.0.0.0/0 X	e.g. SSH for admin desktop

⚠ Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only. X

Add security group rule ↗

- Click Add Security Group Rule



Launch EC2

Add Port Number 4444

▼ Security group rule 2 (TCP, 4444, 0.0.0.0/0)

Type [Info](#)

Custom TCP

Protocol [Info](#)

TCP

Source type [Info](#)

Anywhere

Source [Info](#)

Add CIDR, prefix list or security

0.0.0.0/0 

Port range [Info](#)

4444

[Remove](#)

Description - optional [Info](#)

e.g. SSH for admin desktop

Choose ANYWHERE

 Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only. 



Launch EC2

▼ Configure storage [Info](#) [Advanced](#)

1x GiB gp2 Root volume

ⓘ Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage X

[Add new volume](#)

0 x File systems [Edit](#)

NO CHANGES



Launch EC2

► Advanced details [Info](#) 

- Click Advance Details

Allow tags in metadata [Info](#)

Select ▾

User data [Info](#)

User data has already been base64 encoded



Setup.sh

```
#!/bin/bash
sudo yum update -y &&
sudo amazon-linux-extras install -y docker &&
sudo service docker start &&
sudo usermod -a -G docker ec2-user &&
sudo chkconfig docker on &&
sudo yum install -y git &&
sudo curl -L https://github.com/docker/compose/releases/latest/download/docker-compose-\$\(uname -s\)-\$\(uname -m\) -o /usr/local/bin/docker-compose &&
sudo chmod +x /usr/local/bin/docker-compose &&
docker-compose version &&
sudo curl -L https://cybertek-shared.s3.amazonaws.com/docker-compose.yml -o /home/ec2-user/docker-compose.yml &&
sudo reboot
```



Launch EC2

Paste setup.sh commands here as it is from previous slide



User data [Info](#)

```
#!/bin/bash
sudo yum update -y && sudo amazon-linux-extras install -y docker && sudo
service docker start && sudo usermod -a -G docker ec2-user && sudo chkconfig
docker on && sudo yum install -y git && sudo curl -L
https://github.com/docker/compose/releases/latest/download/docker-
compose-$(uname -s)-$(uname -m) -o /usr/local/bin/docker-compose && sudo
chmod +x /usr/local/bin/docker-compose && docker-compose version && sudo
curl -L https://cybertek-shared.s3.amazonaws.com/docker-compose.yml -o
/home/ec2-user/docker-compose.yml && sudo reboot
```

User data has already been base64 encoded



Launch EC2

▼ Summary

Number of instances [Info](#)
1

Software Image (AMI)
Amazon Linux 2 Kernel 5.10 AMI...[read more](#)
ami-0022f774911c1d690

Virtual server type (instance type)
t2.micro

Firewall (security group)
New security group

Storage (volumes)
1 volume(s) - 8 GiB

ⓘ Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 30 GiB of EBS storage, 2 million IOs, 1 GB of snapshots, and 100 GB of bandwidth to the internet X

Cancel Launch instance



- Launch Instance



Launch EC2

Summary

We noticed that you didn't select a key pair. If you want to be able to connect to your instance it is recommended that you create one.

Key pairs allow you to connect to your instance securely.

Enter the name of the key pair below. When prompted, store the private key in a secure and accessible location on your computer. You will need it later to connect to your instance. [Learn more](#)

Create new key pair Proceed without key pair

Key pair name

The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type
 RSA
RSA encrypted private and public key pair
 ED25519
ED25519 encrypted private and public key pair (Not supported for Windows instances)

Private key file format
 .pem
For use with OpenSSH
 .ppk
For use with PuTTY

- Create new key pair
- Give a key pair name (batchNumber)
- Create key pair



Launch EC2

Create key pair

Key pairs allow you to connect to your instance securely.

Enter the name of the key pair below. When prompted, store the private key in a secure and accessible location on your computer. **You will need it later to connect to your instance.** [Learn more](#)

Key pair name

batchNumbers

The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type

RSA
RSA encrypted private and public key pair

ED25519
ED25519 encrypted private and public key pair (Not supported for Windows instances)

Private key file format

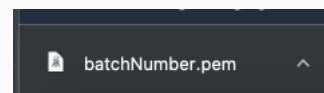
.pem
For use with OpenSSH

.ppk
For use with PuTTY

Create key pair

```
-----BEGIN RSA PRIVATE KEY-----  
MIIEowIBAAKCAQEAmtdTCQj1L8havEBsp5lW+S8MxwAUMsrmvWahsN5dIc9xSe  
CXHstG7IjY/41YYhxnkHi3zMQ8xosCdI4DI5pA/rbzvRjwe9sUeNuZ8NL0hFQtQ  
0UiOvdaq12jodK6pELcoHfLluXNHXff9Hj/xLiKswhUXec3lRTTbiUwxkiVScYrh  
l8JAXr8WsAd4xMvp3Qca81TIXxCi/mTYAGTH0CT/MzSjEMKUhRvkv8x04rChbx4  
D6QN7CseiZu0gEoy5HluRVi3GyS8yTgaedP4Nrf8Pyu0/C0aTq6UEVPl0KWVDiE  
EqJ+SbJ6wEZiDEpXN0mLRp0vkhYFBhqv8awQIDAQABoIBACzG3VVU5d9wXLNV  
eM4zKs/Fdf25TQYPmU43UjVBDg8TY/MM9hpVTMJnfiogxSzImh+PiYp62WUIJawP  
0eAgLvcspI2U EgPAoUMPk5VmviM73HsUbQQuyVu6vtaZEzI7C3fKARw5agYAzQ  
xQLoLk70y/Fas+0cgru28P3mP70xLnhWcx5UUepp+EHvtzm9mqMu6MK3IFtokcy  
hhwfStI3gVwry+rHIJvW4DH+rNGR7HywfGjQ7HJeHbzmb0HLL8pTYjmzI4qFwz7F  
AvGYEkumZCy/Cv99+KP7Bm67j7gJDwnby4R4+o8d34ge8f9hJ8jhPFZjHjvoZEn  
4pmIUoEcgYEAy4ABg+A98ueHyUioEib95eh/uhIDY+w8z8BwigeCYKHv1NQF8  
U6YaQr6d39q/20aExmRKuh3Ej9lJVa08Kob5PCMxyZXka1w8CrMz+pIMzKts  
2uf17YT7dUa/1JmnKGU/T/f6KMGi+0Hub81sjafJUVVxRQTZCpigsWkCgYEaws7D  
K2Amq0/eK6dHnr3pq5YFW0Gmj1k+jj76NeBCKJzuE8vJXmN07me1rnw9GyKoocv  
W4RQ9it4v1N+a1z2+goAbxtdzAMqsqwq5zs55iHD8/KLB2/kBT+iJLZ9DPnCdp  
R4SVNig/RNHH5Kuy192zyWmWuhkTc91N+hk425kCgYBvTRm0RE2MtLW3lS4/n05  
FPaw1tKA8jVZ/884xt5jsF3q2o7Szblh4jEnh3faF6XkdV9rTkMdtAygSbEi7ZW  
3pE/eTK5tlmu1to1WUzt6qV/AtvUenFbRftAqzWDqfBoFudDCcmSJsimrErH4MSK  
mcT+njKVaGT6A0Gv0Ri1IQKBgQCM4919/nyARCawfTmj4F7jk/7wK/h2LHKBJIBd  
WMj+xJ8M9hdDkPoeeZHPYxhTpN1MCcy/f+agTn0+a/x/BuMMdLo4BiKcGKEqx1AF  
C4QRwEFK88AXcr6VESRkrKSZ4vwHtRBv+lQAVf74kJhg6xgGMP2mxzqlqTicIJgIP2  
G7+DGQKBgFnjen90qilTHCgtt2jNGdh2vKQJsb0Hsqifzx89/1kIaeE1XDypcFLi  
4/KTrEsSxkPcOnlgovCzyUmqm1SxDn/D5ICMvJwz1jK/wjh+nzA6bMs0n//m+D  
RXDUH5mycXbAvpUiwoa9v9wdzBnBYRgv012q8fHQ08ixW+q5az1  
-----END RSA PRIVATE KEY-----
```

- It will download it for you



Put this file into Desktop



Launch EC2

EC2 > Instances > Launch an instance



Success

Successfully initiated launch of instance ([i-027850815c3ccaec3](#))

▶ Launch log

Next Steps

Get notified of estimated charges

[Create billing alerts](#) to get an email notification when estimated charges on your AWS bill exceed an amount you define (for example, if you exceed the free usage tier)

How to connect to your instance

Your instance is launching and it might be a few minutes until it is in the running state, when it will be ready for you to use

Click [View Instances](#) to monitor your instance's status. Once your instance is in the 'running' state, you can connect to it from the Instances screen. Find out [how to connect to your instance](#)

[View more resources to get you started](#)

- Click [View All Instances](#)



[View all instances](#)



Launch EC2

Instances (1) Info								C	Connect	Instance state ▾	Actions ▾	Launch instances	▼
<input type="checkbox"/>	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	< 1 >				
<input type="checkbox"/>	selenium-grid...	i-027850815c3ccaec3	Running		t2.micro	Initializing	No alarms		us-east-1d	ec2-54-80-152-1.c			



Wait for 2/2 checks passed

Instances (1) Info								C	Connect	Instance state ▾	Actions ▾	Launch instances	▼
<input type="checkbox"/>	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	< 1 >				
<input type="checkbox"/>	selenium-grid...	i-027850815c3ccaec3	Running		2/2 checks passed	No alarms		us-east-1d	ec2-54-80-152-1.c				



READY TO USE



CYDEO

CONNECTION



Connection

- Click CONNECT

A red arrow pointing upwards from the 'Select Server' instruction to the first row of the table, which contains the selected server information.

Instances (1/1) Info		C	Connect	Instance state	Actions	Launch instances	▼
		Search		< 1 >	⚙️		
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
<input checked="" type="checkbox"/> selenium-grid-...	i-027850815c3ccaec3	Running	t2.micro	2/2 checks passed	No alarms	us-east-1d	ec2-54-80-152-1.c...

- Select Server



Connection

- Select EC2 Instance Connect

Connect to instance [Info](#)

Connect to your instance i-027850815c3ccaec3 (selenium-grid-server) using any of these options

EC2 Instance Connect [Session Manager](#) [SSH client](#) [EC2 Serial Console](#)

Instance ID
 [i-027850815c3ccaec3 \(selenium-grid-server\)](#)

Public IP address
 [54.80.152.1](#)

User name

Connect using a custom user name, or use the default user name ec2-user for the AMI used to launch the instance.

i **Note:** In most cases, the guessed user name is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI user name.

[Cancel](#) [Connect](#)

- Click CONNECT



Connection

- Welcome to Linux SERVER



i-027850815c3ccaec3 (selenium-grid-server)

Public IPs: 54.80.152.1 Private IPs: 172.31.19.212



Run Docker Containers

- Type here **docker -v** to see version

```
  _|_(_|_) /   Amazon Linux 2 AMI  
__|_\_|_|  
https://aws.amazon.com/amazon-linux-2/  
[ec2-user@ip-172-31-19-212 ~]$ docker -v  
Docker version 20.10.13, build a224086  
[ec2-user@ip-172-31-19-212 ~]$ █
```

- Type here **ls** to see docker file

```
https://aws.amazon.com/amazon-linux-2/  
[ec2-user@ip-172-31-19-212 ~]$ ls  
docker-compose.yml  
[ec2-user@ip-172-31-19-212 ~]$ █
```



Run Selenium Grid

- docker-compose up -d

```
[ec2-user@ip-172-31-19-212 ~]$ docker-compose up -d
[+] Running 3/3
# Network ec2-user_default    Created
# Container ec2-user-hub-1    Started
# Container ec2-user-chrome-1 Started
```



```
... tab0c4914688 Full complete
[+] Running 3/3
# Network ec2-user_default    Created
# Container ec2-user-hub-1    Started
# Container ec2-user-chrome-1 Started
[ec2-user@ip-172-31-19-212 ~]$ █
```



Check Grid Console

`http://server-ip:4444/grid/console`



Stop Selenium Grid

- docker-compose down

```
ec2-user@ip-172-31-19-212 ~]$ docker-compose down
+] Running 3/2
  ⚡ Container ec2-user-chrome-1    Removed
  ⚡ Container ec2-user-hub-1      Removed
  ⚡ Network ec2-user_default     Removed
ec2-user@ip-172-31-19-212 ~]$ █
```



Driver Setup

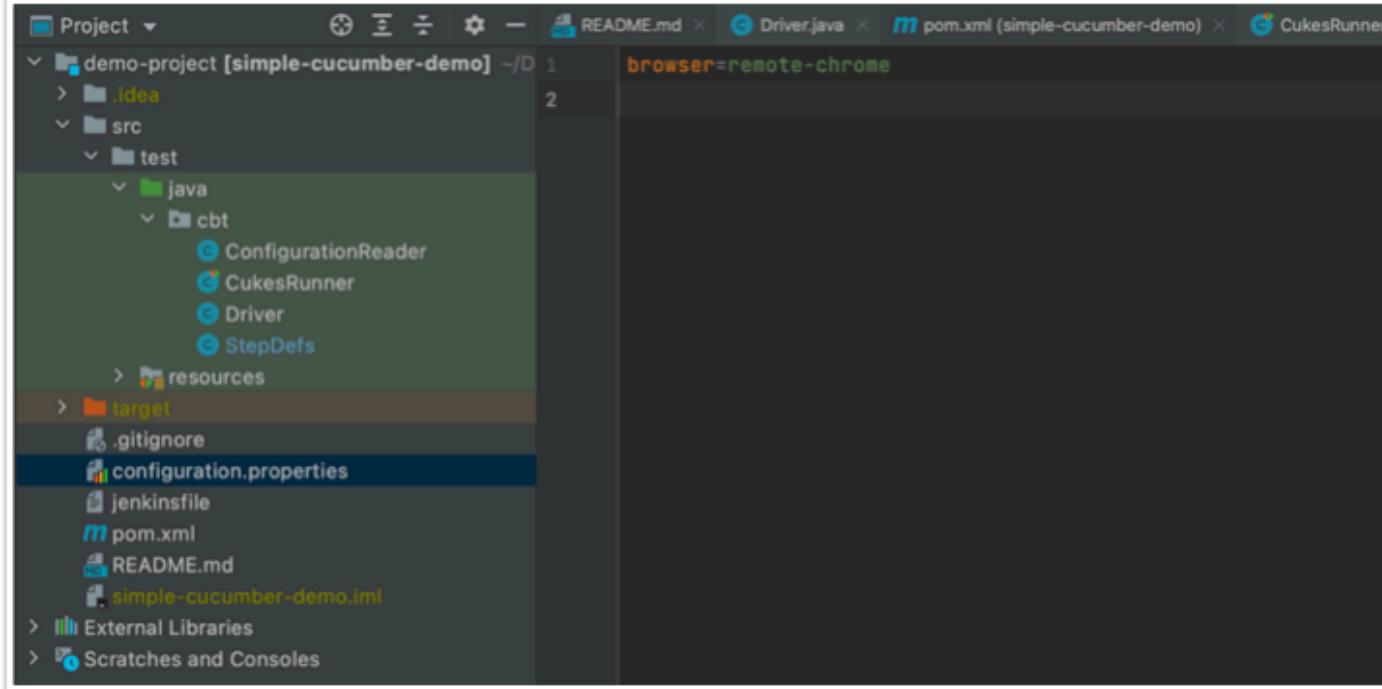
Add new case for remote-chrome and use your gridAddress from EC2

```
switch (browser) {  
    case "remote-chrome":  
        try {  
            // assign your grid server address  
            String gridAddress = "54.235.53.73";  
            URL url = new URL(spec: "http://" + gridAddress + ":4444/wd/hub");  
            DesiredCapabilities desiredCapabilities = new DesiredCapabilities();  
            desiredCapabilities.setBrowserName("chrome");  
            driver = new RemoteWebDriver(url, desiredCapabilities);  
        } catch (Exception e) {  
            e.printStackTrace();  
        }  
        break;  
}
```



Driver Setup

Update properties file to use remote-chrome



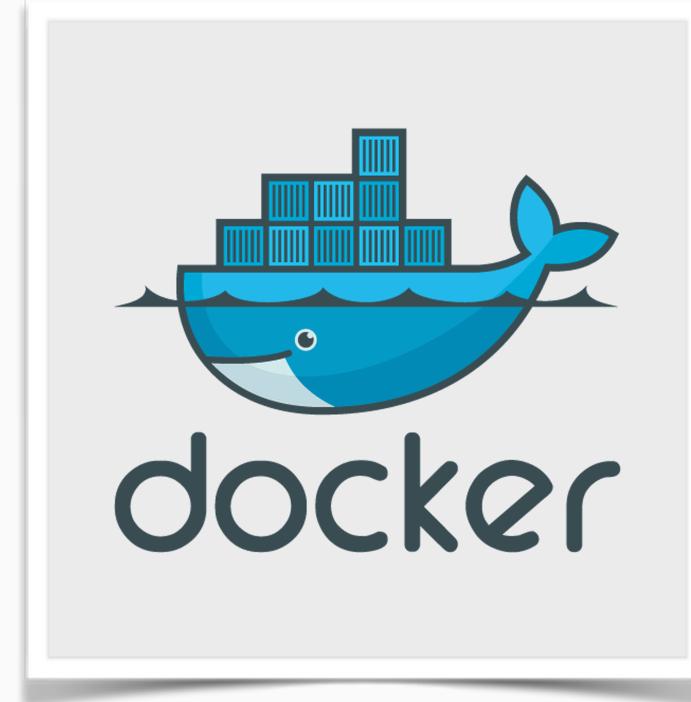
CYDEO

DOCKER



Docker

Docker is a tool designed to make it easier to create, deploy, and run applications by using containers. Containers allow a developer to package up an application with all of the parts it needs, such as libraries and other dependencies, and deploy it as one package. By doing so, thanks to the container, the developer can rest assured that the application will run on any other Linux machine regardless of any customized settings that machine might have that could differ from the machine used for writing and testing the code.



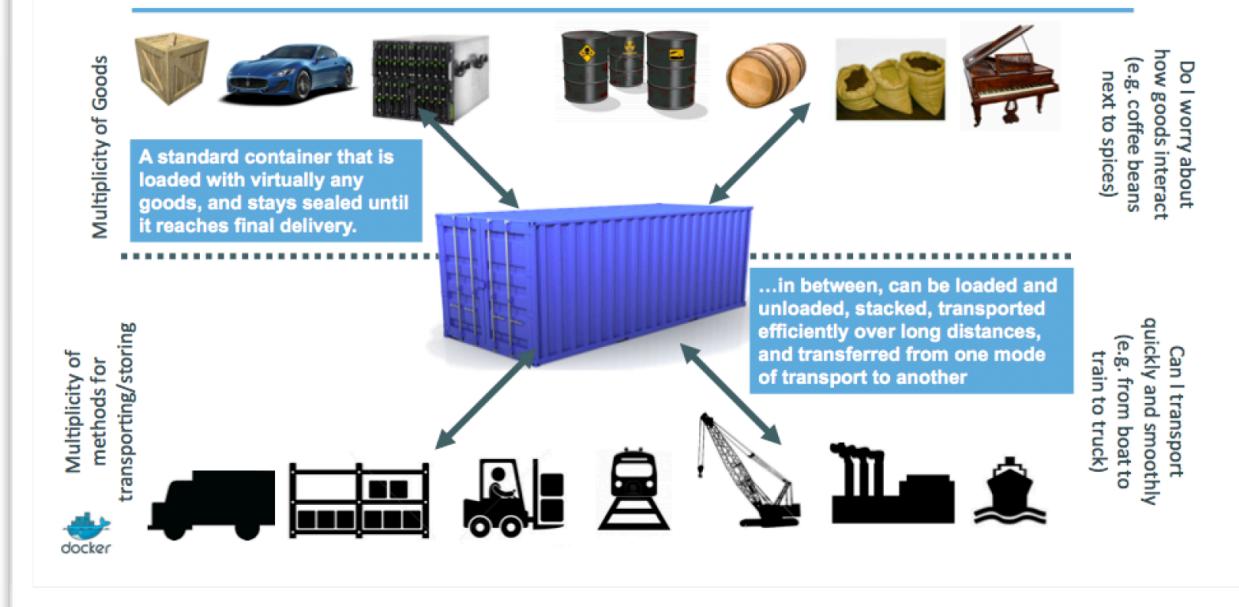
Docker

In simpler words, 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. The key benefit of Docker is that it allows users to **package an application with all of its dependencies into a standardized unit** for software development. Unlike virtual machines, containers do not have high overhead and hence enable more efficient usage of the underlying system and resources.



Docker

Intermodal shipping containers



Docker

This spawned a Shipping Container Ecosystem!

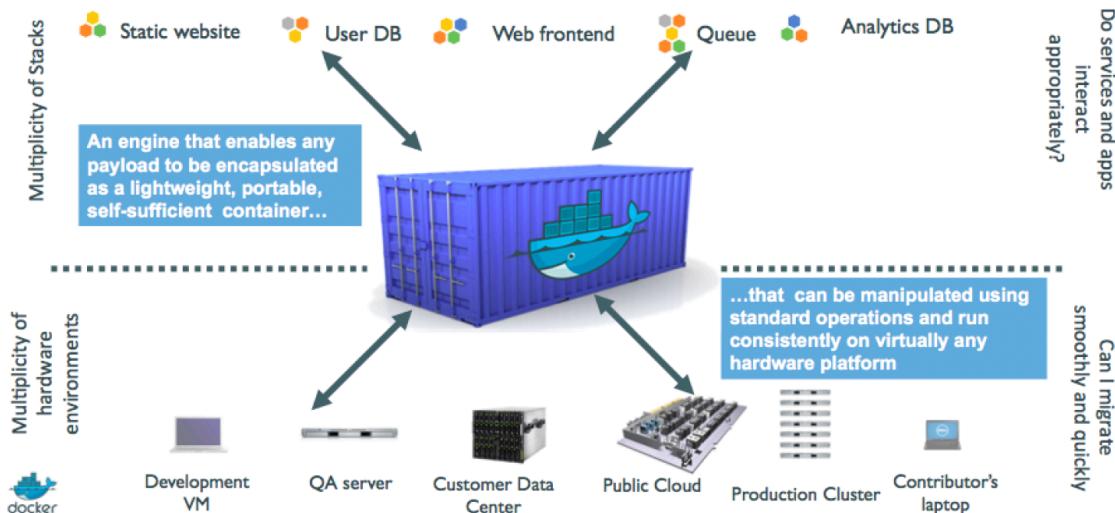


- 90% of all cargo now shipped in a standard container
- Order of magnitude reduction in cost and time to load and unload ships
- Massive reduction in losses due to theft or damage
- Huge reduction in freight cost as percent of final goods (from >25% to <3%)
→ massive globalization
- 5000 ships deliver 200M containers per year



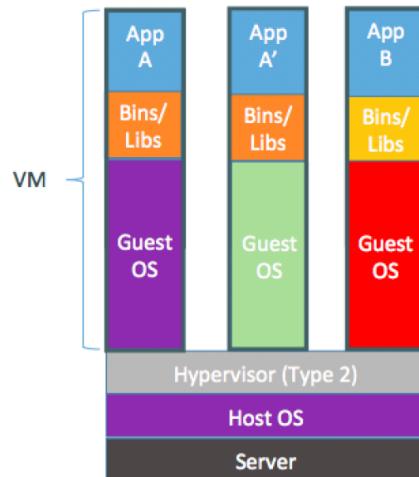
Docker

A shipping container system for applications



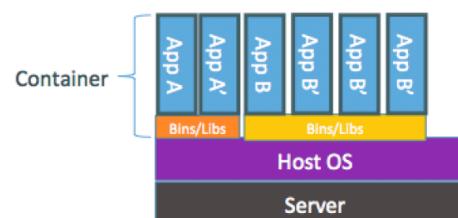
Docker

Step 1: containers as lightweight VMs



Containers are isolated,
but share OS kernel and, where
appropriate, bins/libraries

...result is significantly faster deployment,
much less overhead, easier migration,
faster restart



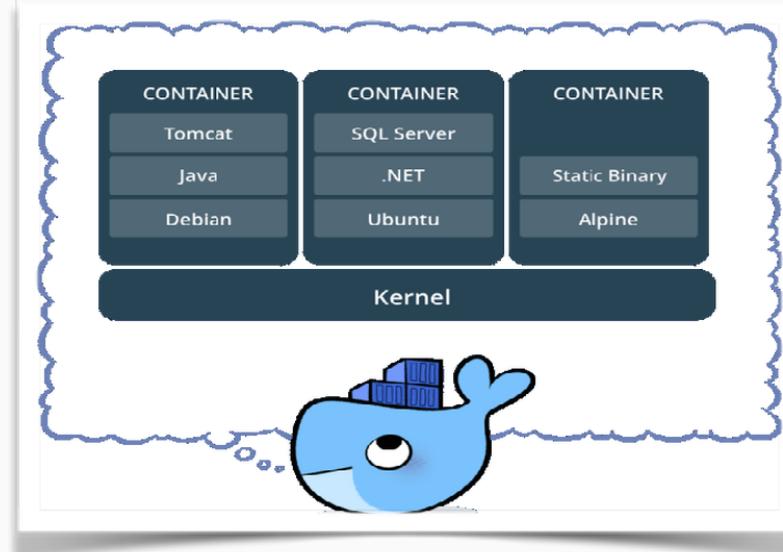
VMs vs Containers

VMs	Containers
Heavyweight	Lightweight
Limited performance	Native performance
Each VM runs in its own OS	All containers share the host OS
Hardware-level virtualization	OS virtualization
Startup time in minutes	Startup time in milliseconds
Allocates required memory	Requires less memory space
Fully isolated and hence more secure	Process-level isolation, possibly less secure



Docker Container

- **Container** is built from an **image**.
- A container consists of an operating system, user-added **files**, and **meta-data**.
- That image tells Docker what the container **holds**, what process to **run** when the container is launched, and a variety of other **configuration** data.
- The Docker image is **read-only**.
- When Docker runs a container from an image, it adds a read-write layer on top of the image (using a union file system as we saw earlier) in which your application can then run.



Docker Compose

Compose is a tool for defining and running multi-container Docker applications. With Compose, you use a YAML file to configure your application's services.

Then, with a single command, you create and start all the services from your configuration. To learn more about all the features of Compose, see the list of features.

Compose works in all environments: production, staging, development, testing, as well as CI workflows. You can learn more about each case in Common Use Cases.

```
version: '2.0'
services:
  web:
    build: .
    ports:
      - "5000:5000"
    volumes:
      - .:/code
      - logvolume01:/var/log
    links:
      - redis
  redis:
    image: redis
volumes:
  logvolume01: {}
```



Some Docker Commands

```
docker build -t friendlyname . # Create image using this directory's Dockerfile
docker run -p 4000:80 friendlyname # Run "friendlyname" mapping port 4000 to 80
docker run -d -p 4000:80 friendlyname      # Same thing, but in detached mode
docker ps                         # See a list of all running containers
docker stop <hash>                # Gracefully stop the specified container
docker ps -a                      # See a list of all containers, even the ones not running
docker kill <hash>                # Force shutdown of the specified container
docker rm <hash>                  # Remove the specified container from this machine
docker rm $(docker ps -a -q)       # Remove all containers from this machine
docker images -a                  # Show all images on this machine
docker rmi <imagename>           # Remove the specified image from this machine
docker rmi $(docker images -q)    # Remove all images from this machine
docker login                      # Log in this CLI session using your Docker credentials
docker tag <image> username/repository:tag # Tag <image> for upload to registry
docker push username/repository:tag # Upload tagged image to registry
docker run username/repository:tag # Run image from a registry
```

