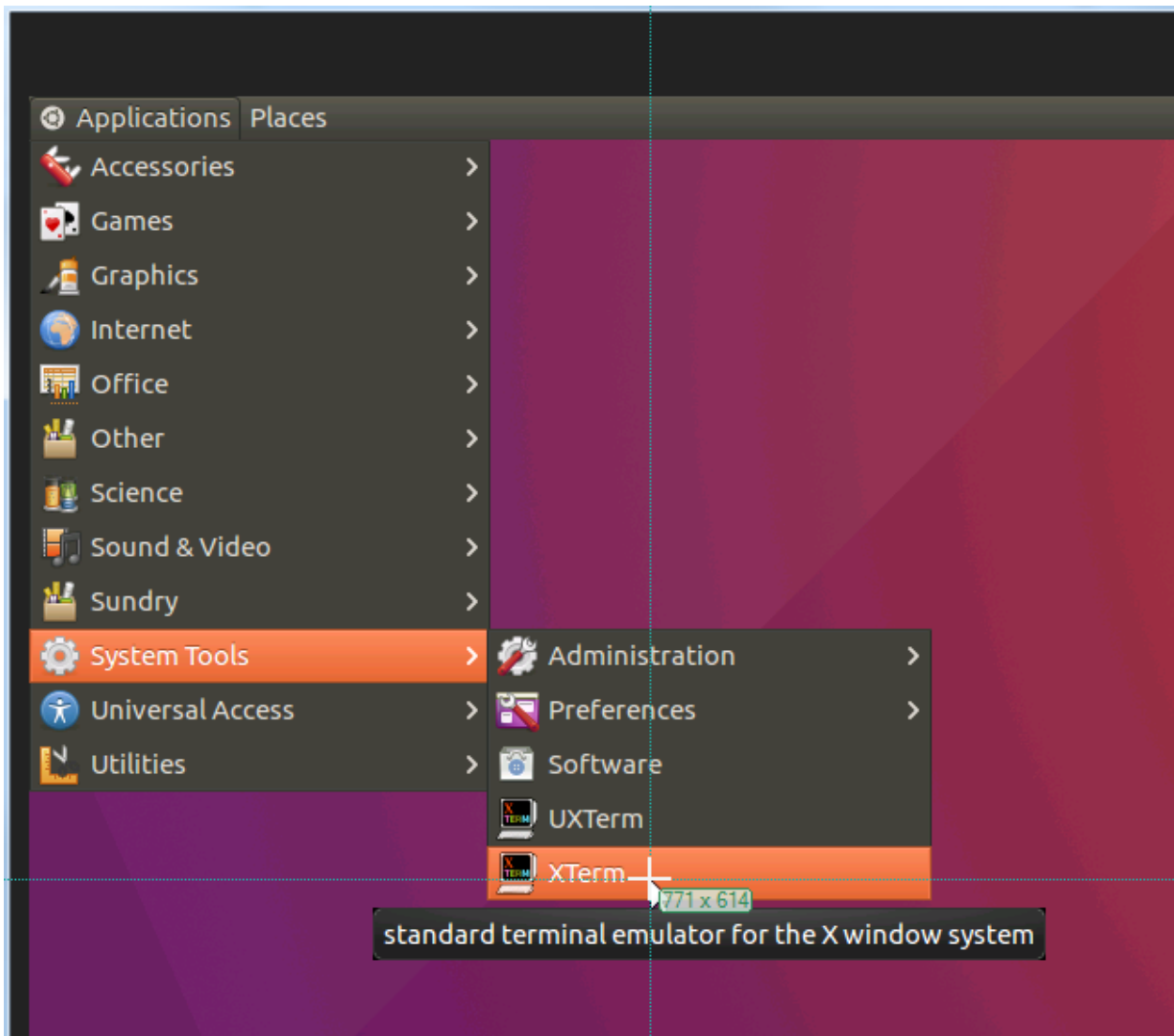


Install Terminal:

1.

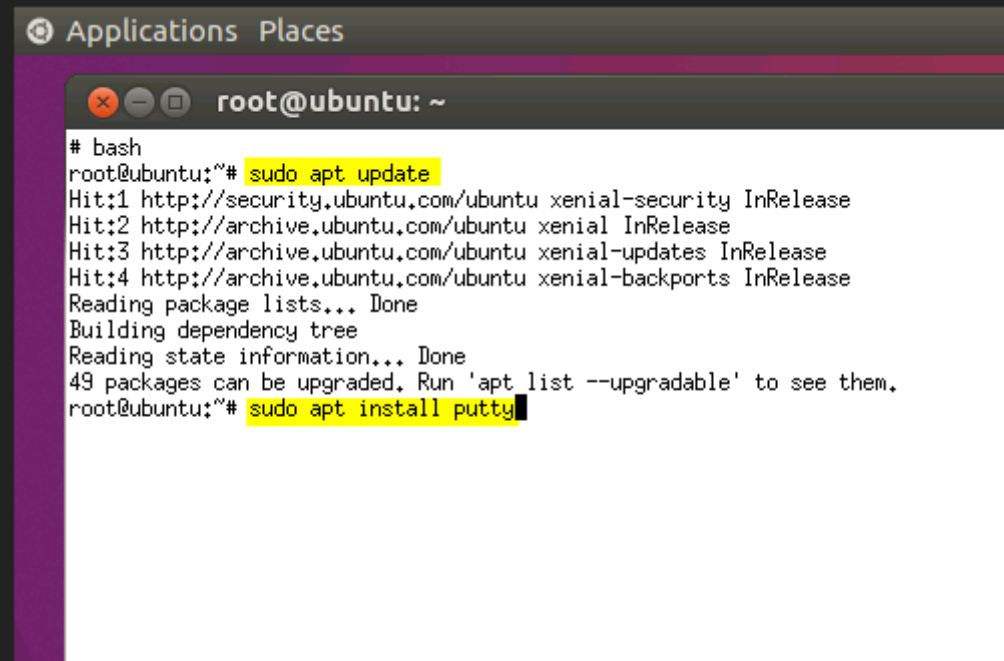


2. `ps aux | grep -i apt`

***Find the pid and kill it with "sudo kill -9 pid"

***You can ignore the line contain "grep --color=auto"

3. .



```
Applications Places
root@ubuntu: ~
# bash
root@ubuntu:~# sudo apt update
Hit:1 http://security.ubuntu.com/ubuntu xenial-security InRelease
Hit:2 http://archive.ubuntu.com/ubuntu xenial InRelease
Hit:3 http://archive.ubuntu.com/ubuntu xenial-updates InRelease
Hit:4 http://archive.ubuntu.com/ubuntu xenial-backports InRelease
Reading package lists... Done
Building dependency tree
Reading state information... Done
49 packages can be upgraded. Run 'apt list --upgradable' to see them.
root@ubuntu:~# sudo apt install putty
```

Getting started with Kafka lab

Let's show a simple example using producers and consumers from the Kafka command line.

Download Kafka 0.10.2.x from the [Kafka download page](#). Later versions will likely work, but this was example was done with 0.10.2.x. (Download Link:

http://apache.claz.org/kafka/0.10.2.2/kafka_2.11-010.2.2.tgz)

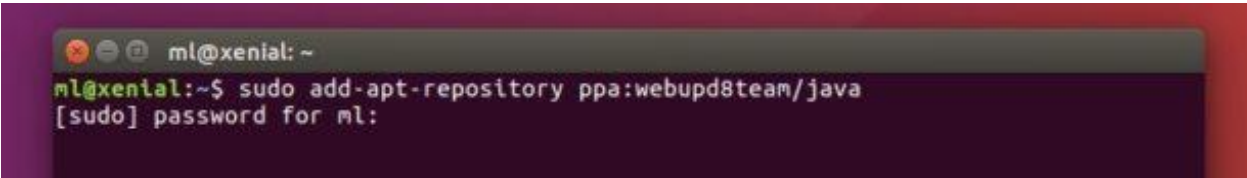
Install Java 8

1. Add the PPA.

Open terminal (Ctrl+Alt+T) and run the command:

```
sudo add-apt-repository ppa:webupd8team/java
```

Type in your password when it asks and hit Enter.



```
ml@xenial: ~
ml@xenial:~$ sudo add-apt-repository ppa:webupd8team/java
[sudo] password for ml:
```

2. Update and install the installer script:

Run commands to update system package index and install Java installer script:

```
sudo apt update; sudo apt install oracle-java8-installer
```

You may replace `oracle-java8-installer` with **`oracle-java9-installer`** to install Java 9.

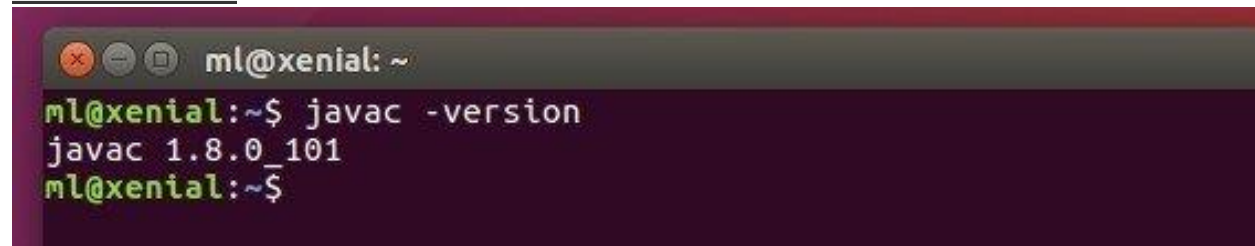
While the install process, you have to accept Java license to continue downloading & installing Java binaries.



3. Check the Java version

To check the Java version after installing the package, run command:

```
javac -version
```



4. Set Java environment variables

The PPA also contains a package to automatically set Java environment variables, just run command:

```
sudo apt install oracle-java8-set-default
```

For Java 9, install the package **`oracle-java9-set-default`** instead.

We unzipped the Kafka download and put it in `~/kafka-training/`, and then renamed the Kafka install folder to `kafka`. Please do the same.

1. `mkdir ~/kafka-training/`
2. `cd ~/kafka-training/`
3. `wget http://apache.claz.org/kafka/0.10.2.2/kafka_2.11-010.2.2.tgz`

4. `tar xvzf kafka_2.11-0.10.2.2.tgz`
5. `mv kafka_2.11-0.10.2.2 kafka`

Next, we are going to run *ZooKeeper* and then run *Kafka Server/Broker*. We will use some Kafka command line utilities, to create Kafka topics, send messages via a producer and consume messages from the command line.

Run ZooKeeper for Kafka

Kafka relies on ZooKeeper. To keep things simple, we will use a single ZooKeeper node.

Kafka provides a startup script for ZooKeeper called `zookeeper-server-start.sh` which is located at `~/kafka-training/kafka/bin/zookeeper-server-start.sh`.

The Kafka distribution also provide a ZooKeeper config file which is setup to run single node.

To run ZooKeeper, we create this script in `kafka-training` and run it.

~/kafka-training/run-zookeeper.sh

```
#!/usr/bin/env bash
cd ~/kafka-training

kafka/bin/zookeeper-server-start.sh \
  kafka/config/zookeeper.properties
```

Run run-zookeeper.sh

```
cd ~/kafka-training

chmod 755 ./run-zookeeper.sh
$ ./run-zookeeper.sh
```

Wait about 30 seconds or so for ZooKeeper to startup.

Run Kafka Server

Kafka also provides a startup script for the Kafka server called `kafka-server-start.sh` which is located at `~/kafka-training/kafka/bin/kafka-server-start.sh`.

The Kafka distribution also provide a Kafka config file which is setup to run Kafka single node, and points to ZooKeeper running on `localhost:2181`.

To run Kafka, we create this script in `kafka-training` and run it in another terminal window.

~/kafka-training/run-kafka.sh

```
#!/usr/bin/env bash
cd ~/kafka-training

kafka/bin/kafka-server-start.sh \
    kafka/config/server.properties
```

Run run-kafka.sh

```
~/kafka-training
chmod 755 ./run-kafka.sh
$ ./run-kafka.sh
```

Wait about 30 seconds or so for Kafka to startup.

Now let's create the topic that we will send records on.

Create Kafka Topic

Kafka also provides a utility to work with topics called `kafka-topics.sh` which is located at `~/kafka-training/kafka/bin/kafka-topics.sh`.

We will use this tool to create a topic called `my-topic` with a replication factor of 1 since we only have one server. We will use thirteen partitions for `my-topic`, which means we could have up to 13 Kafka consumers.

To run Kafka, create this script in `kafka-training/lab1`, and run it in another terminal window.

~/kafka-training/lab1/create-topic.sh

```
#!/usr/bin/env bash

cd ~/kafka-training

# Create a topic
kafka/bin/kafka-topics.sh --create \
  --zookeeper localhost:2181 \
  --replication-factor 1 --partitions 13 \
  --topic my-topic
```

Run create-topic.sh

```
~/kafka-training/lab1
```

```
$ ./create-topic.sh
```

```
Created topic "my-topic".
```

Notice we created a topic called `my-topic`.

List Topics

You can see which topics that Kafka is managing using `kafka-topics.sh` as follows.

Create the file in `~/kafka-training/lab1/list-topics.sh`. and run it.

~/kafka-training/lab1/list-topics.sh

```
#!/usr/bin/env bash

cd ~/kafka-training

# List existing topics
kafka/bin/kafka-topics.sh --list \
```

```
--zookeeper localhost:2181
```

Notice that we have to specify the location of the ZooKeeper cluster node which is running on `localhost` port `2181`.

Run list-topics.sh

```
~/kafka-training/lab1
$ ./list-topics.sh
__consumer_offsets
__schemas
my-example-topic
my-example-topic2
my-topic
new-employees
```

You can see the topic `my-topic` in the list of topics.

Run Kafka Producer Console

The Kafka distribution provides a command utility to send messages from the command line. It starts up a terminal window where everything you type is sent to the Kafka topic.

Kafka provides the utility `kafka-console-producer.sh` which is located at `~/kafka-training/kafka/bin/kafka-console-producer.sh` to send messages to a topic on the command line.

Create the file in `~/kafka-training/lab1/start-producer-console.sh` and run it.

~/kafka-training/lab1/start-producer-console.sh

```
#!/usr/bin/env bash
cd ~/kafka-training
```

```
kafka/bin/kafka-console-producer.sh \
```

```
--broker-list localhost:9092 \  
--topic my-topic
```

Notice that we specify the Kafka node which is running at `localhost:9092`.

Run start-producer-console.sh and send at least four messages

```
~/kafka-training/lab1  
$ ./start-producer-console.sh  
This is message 1  
This is message 2  
This is message 3  
Message 4  
Message 5
```

In order to see these messages, we will need to run the consumer console.

Run Kafka Consumer Console

The Kafka distribution provides a command utility to see messages from the command line. It displays the messages in various modes.

Kafka provides the utility `kafka-console-consumer.sh` which is located at `~/kafka-training/kafka/bin/kafka-console-producer.sh` to receive messages from a topic on the command line.

Create the file in `~/kafka-training/lab1/start-consumer-console.sh` and run it.

~/kafka-training/lab1/start-producer-console.sh

```
#!/usr/bin/env bash  
cd ~/kafka-training  
  
kafka/bin/kafka-console-consumer.sh \  
--bootstrap-server localhost:9092 \  

```



```
--topic my-topic \  
--from-beginning
```

Notice that we specify the Kafka node which is running at `localhost:9092` like we did before, but we also specify to read all of the messages from `my-topic` from the beginning `--from-beginning`.

Run `start-consumer-console.sh` in another terminal

```
~/kafka-training/lab1  
$ ./start-consumer-console.sh  
Message 4  
This is message 2  
This is message 1  
This is message 3  
Message 5  
Message 6  
Message 7
```

Notice that the messages are not coming in order. This is because we only have one consumer so it is reading the messages from all 13 partitions. Order is only guaranteed within a partition.

Review of using Kafka from the command line

What server do you run first?

You need to run ZooKeeper than Kafka.

What tool do you use to create a topic?

`kafka-topics.sh`

What tool do you use to see topics?

`kafka-topics.sh`

What tool did we use to send messages on the command line?

kafka-console-producer.sh

What tool did we use to view messages in a topic?

kafka-console-consumer.sh

Why were the messages coming out of order?

The messages were being sharded among 13 partitions.

How could we get the messages to come in order from the consumer?

We could use only one partition or start up 13 consumers.