HyperLedger Fabric Installation Guide with CODE

For our Hyperledger fabric network to be made we need some prerequisites to be installed. And also note that this network can only be developed in linux systems. It does not support for the Windows and IOS.

The prerequisites are :

- Docker Engine and Docker Compose
- Nodejs and NPM
- Git
- Python 2.7.x

To run Hyperledger Composer and Hyperledger Fabric, it is recommend you have at least 4Gb of memory.
If installing Hyperledger Composer using Linux, be aware of the following advice:

- Login as a normal user, rather than root.
- Do not su to root.
- When installing prerequisites, use curl, then unzip using sudo.
- Run prereqs-ubuntu.sh as a normal user. It may prompt for root password as some of it's actions are required to be run as root.
- Do not use npm with sudo or su to root to use it.
- Avoid installing node globally as root.

If you're running on Ubuntu, you can download the prerequisites using the following commands:

```
curl -O https://hyperledger.github.io/composer/latest/prereqs-ubuntu.sh
```

```
chmod u+x prereqs-ubuntu.sh
```

Next run the script - as this briefly uses sudo during its execution, you will be prompted for your password.

```
./prereqs-ubuntu.sh
```

User must then logout and login upon completion of script
Update package lists

```
echo "# Updating package lists"
```

```
sudo apt-add-repository -y ppa:git-core/ppa
```

```
sudo apt-get update
```

Install Git

```
echo "# Installing Git"

sudo apt-get install -y git
```

Install nvm dependencies

```
echo "# Installing nvm dependencies"

sudo apt-get -y install build-essential libssl-dev
```

Execute nvm installation script

```
echo "# Executing nvm installation script"

curl -
o- https://raw.githubusercontent.com/creationix/nvm/v0.33.2/in
stall.sh | bash
```

Set up nvm environment without restarting the shell

```
export NVM_DIR="${HOME}/.nvm"

[ -s "${NVM_DIR}/nvm.sh" ] && . "${NVM_DIR}/nvm.sh"

[ -
s "${NVM_DIR}/bash_completion" ] && . "${NVM_DIR}/bash_complet
ion"
```

Install node

```
echo "# Installing nodeJS"

nvm install 8

nvm use 8
```

Ensure that CA certificates are installed

```
sudo apt-get -y install apt-transport-https ca-certificates
```

Add Docker repository key to APT keychain

```
curl -
fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-
key add -
```

Update where APT will search for Docker Packages

```
echo "deb [arch=amd64] https://download.docker.com/linux/ubuntu ${CODENAME} stable" | \
    sudo tee /etc/apt/sources.list.d/docker.list
```

Update package lists

```
sudo apt-get update
```

Verifies APT is pulling from the correct Repository

```
sudo apt-cache policy docker-ce
```

Install kernel packages which allows us to use aufs storage driver if V14 (trusty/utopic)

```
if [ "${CODENAME}" == "trusty" ]; then

    echo "# Installing required kernel packages"

    sudo apt-get -y install linux-image-extra-$(uname -r) linux-image-extra-virtual
```

Install Docker

```
echo "# Installing Docker"

sudo apt-get -y install docker-ce
```

Add user account to the docker group

```
sudo usermod -aG docker $(whoami)
```

Install docker compose

```
echo "# Installing Docker-Compose"

sudo curl -L "https://github.com/docker/compose/releases/download/1.13.0/docker-compose-$(uname -s)-$(uname -m)" \

    -o /usr/local/bin/docker-compose

sudo chmod +x /usr/local/bin/docker-compose
```

Install python v2 if required

```
set +e

COUNT="$(python -V 2>&1 | grep -c 2.)"

if [ ${COUNT} -ne 1 ]
```

```
then
   sudo apt-get install -y python-minimal
```

Install unzip, required to install hyperledger fabric.

```
sudo apt-get -y install unzip
```

Print installation details for user

```
echo ''
echo 'Installation completed, versions installed are:'
echo ''
echo -n 'Node:           '
node --version
echo -n 'npm:            '
npm --version
echo -n 'Docker:         '
docker --version
echo -n 'Docker Compose: '
docker-compose --version
echo -n 'Python:         '
python3 -V
```

Print reminder of need to logout in order for these changes to take effect!

```
echo ''
echo "Please logout then login before continuing."
```

composer-cli is the only essential package. The rest aren't core components but will turn out to be extremely useful over time. We will learn more about what each of these do as we come across them.
We then specify the version of Fabric we want, at the time of writing we need 1.2, hence **hlfv12**. Then, we download the fabric runtime and start it up. Basically what we did here was just download and start a local Fabric network. We can stop is using

```
./stopFabric.sh
```

if we want to. At the end of our development session, we should run

```
./teardownFabric.sh
```

Open terminal in a directory of choice and type

`yo hyperledger-composer`

Select Business Network and name it according to your network.

Add the codes as below into their respective files:

org.example.biznet.cto:

```
namespace org.example.biznet
/**
 * THe participant model for a Trader
 */
participant Trader identified by traderId {
  o String traderId
  o String traderName
}

/**
 * The asset participants will be trading.
 * Each card has certain properties such as name,
 * description, type, and quantity which can
 * be used for the frontend application
 */
asset TradingCard identified by cardId {
  o String cardId
  o String cardName
  o String cardDescription
  o GameType cardType default="Baseball" // If no value is provided, it takes
the default value
  o Double quantity
  o Double price default=10.0
  o Boolean forTrade
  --> Trader owner
}

/**
 * A transaction which allows Traders to buy other
 * Traders' cards if they're available for trade
 */
transaction TradeCard {
  --> TradingCard card
  --> Trader newOwner
}

/**
 * A notification event to be emitted whenever
```

```
 * any card is traded
 */
event TradeNotification {
  --> TradingCard card
}

/**
 * Enumerated types are used to specify a type
 * which can have 1 or N possible values, and nothing else.
 */
enum GameType {
  o Baseball
  o Football
  o Cricket
}
```

logic.js:

```
/**
 * Buy card transaction
 * @param {org.example.biznet.TradeCard} trade
 * @transaction
 */
async function buyCard(trade) {
  if (trade.card.forTrade) {
    // If card is available for trade
    trade.card.owner = trade.newOwner;
    return getAssetRegistry("org.example.biznet.TradingCard")
      .then(assetRegistry => {
        return assetRegistry.update(trade.card); // Update the network registr
y
      })
      .then(() => {
        let event = getFactory().newEvent(
          "org.example.biznet",
          "TradeNotification"
        ); // Get a reference to the event specified in the modeling language
        event.card = trade.card;
        emit(event); // Fire off the event
      });
  }
}
```

permissions.acl:

```
rule NetworkAdminUser {
    description: "Grant business network administrators full access to user re
sources"
    participant: "org.hyperledger.composer.system.NetworkAdmin"
    operation: ALL
    resource: "**"
    action: ALLOW
}

rule NetworkAdminSystem {
    description: "Grant business network administrators full access to system
resources"
    participant: "org.hyperledger.composer.system.NetworkAdmin"
    operation: ALL
    resource: "org.hyperledger.composer.system.**"
    action: ALLOW
}

rule AllParticipantsHaveAccessToAllResources {
  description: "Allow all participants to have access to all resources and mak
e transactions"
  participant: "ANY"
  operation: ALL
  resource: "org.example.biznet.*"
  action: ALLOW
}
```

Now that all the coding is done, it's time to make an archive file for our business network so we can deploy it on our local Fabric runtime. To do this, open Terminal in your project directory and type this:

```
composer archive create --sourceType dir --sourceName
```

We can install and deploy the network to our local Fabric runtime using the PeerAdmin user. To install the business network, type

```
composer network install --archiveFile filename@0.0.1.bna --
card PeerAdmin@hlfv1
```

To deploy the business network, type

```
composer network start --networkName cards-trading-network --networkVersion 0.0.1 --networkAdmin admin --networkAdminEnrollSecret adminpw --card PeerAdmin@hlfv1 --file filename.card
```

The networkName and networkVersion must be the same as specified in your package.json otherwise it won't work. File takes the name of the file to be created for THIS network's business card. This card then needs to be imported to be usable by typing

```
composer card import --file filename.card
```

Amazing. We can now confirm that our network is up and running by typing

```
composer network ping --card admin@filename
```

card this time takes the admin card of the network we want to ping.If everything went well, you should see something similar to this:
Your network version will be 0.0.1 or whatever your package.json specifies — I actually forgot to take this screenshot and uploaded it after I was done writing the tutorial and making edits
Now that our network is up and running on Fabric, we can start Composer Playground to interact with it. To do this, type

```
composer-playground
```

This opens up our ledger fabric.