**Assignment 6**

**Create Business Network Using Hyperledger Composer**

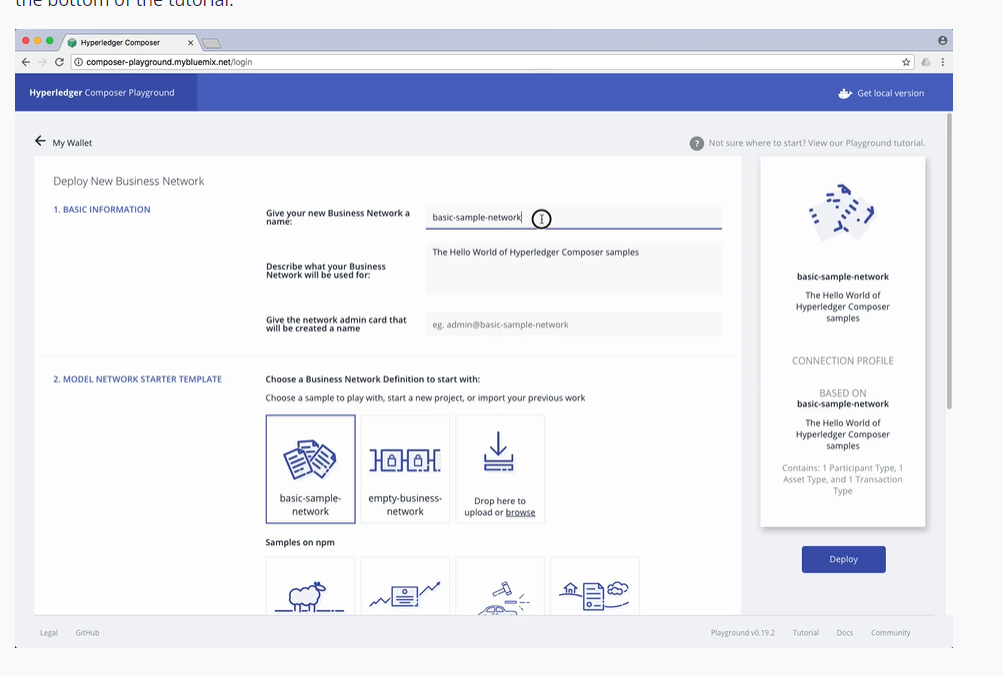
Open [Composer Playground](https://composer-playground.mybluemix.net/) (note, this link will take you to the web Composer Playground - you can also follow along in a local version if you've already installed the development environment).

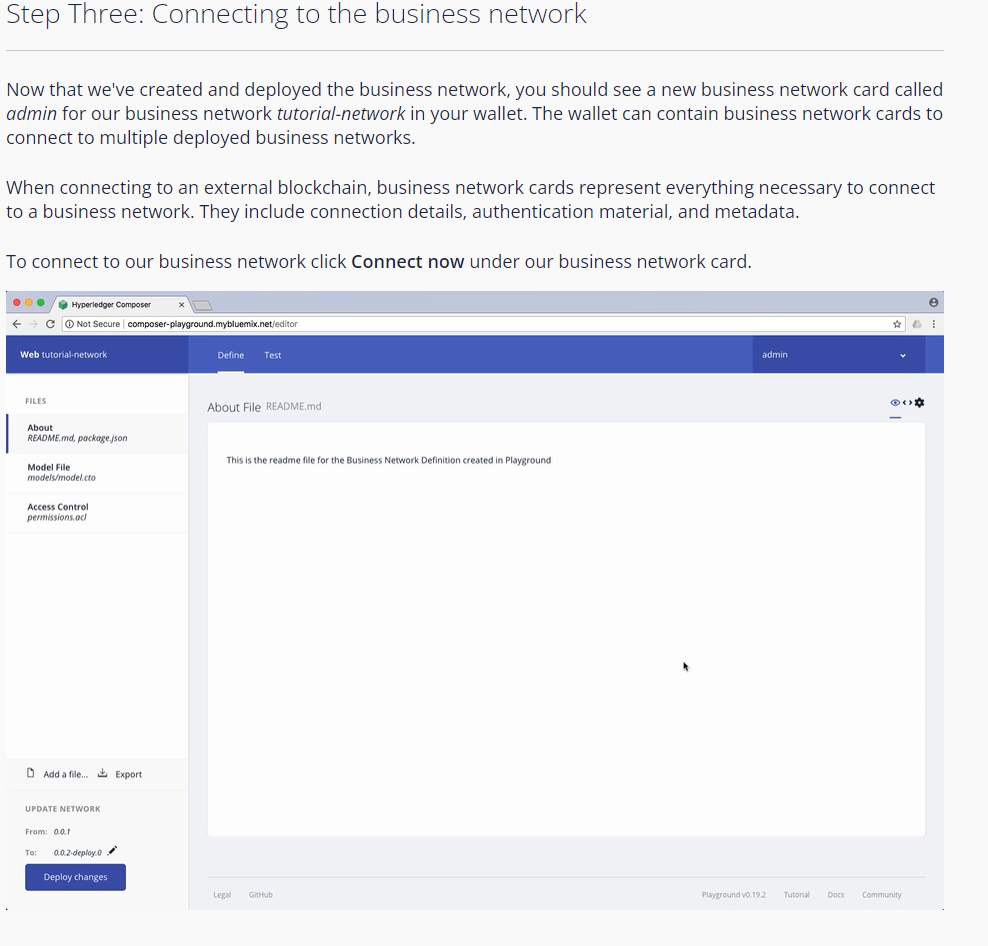
You should see the My Business Networks screen. The My Business Networks page shows you a summary of the business networks you can connect to, and the identities you can use to connect to them. Don't worry about this too much for the time being, as we're going to create our own network.

Creating a new business network

Next, we want to create a new business network from scratch. A business network has a couple of defining properties; a name, and an optional description. You can also choose to base a new business network on an existing template, or import your own template.

1. Click **Deploy a new business network** under the Web Browser heading to get started.
2. The new business network needs a name, let's call it tutorial-network.
3. Optionally, you can enter a description for your business network.
4. Next we must select a business network to base ours on, because we want to build the network from scratch, click **empty-business-network**.
5. Now that our network is defined, click **Deploy**.





As you can see, we're in the **Define** tab right now, this tab is where you create and edit the files that make up a business network definition, before deploying them and testing them using the **Test** tab.

As we selected an empty business network template, we need to modify the template files provided. The first step is to update the model file. Model files define the assets, participants, transactions, and events in our business network.

For more information on our modeling language, check our [documentation](https://hyperledger.github.io/composer/v0.19/reference/cto_language.html).

Click the **Model file** to view it.

1. Delete the lines of code in the model file and replace it with this:

/\*\*

\* My commodity trading network

\*/

namespace org.example.mynetwork

asset Commodity identified by tradingSymbol {

o String tradingSymbol

o String description

o String mainExchange

o Double quantity

--> Trader owner

}

participant Trader identified by tradeId {

o String tradeId

o String firstName

o String lastName

}

transaction Trade {

--> Commodity commodity

--> Trader newOwner

}

This domain model defines a single asset type Commodity and single participant type Trader and a single transaction type Trade that is used to modify the owner of a commodity.

Now that the domain model has been defined, we can define the transaction logic for the business network. Composer expresses the logic for a business network using JavaScript functions. These functions are automatically executed when a transaction is submitted for processing.

For more information on writing transaction processor functions, check our [documentation](https://hyperledger.github.io/composer/v0.19/reference/js_scripts.html).

1. Click the **Add a file** button.
2. Click the **Script file** and click **Add**.
3. Delete the lines of code in the script file and replace it with the following code:

Copy

/\*\*

\* Track the trade of a commodity from one trader to another

\* @param {org.example.mynetwork.Trade} trade - the trade to be processed

\* @transaction

\*/

async function tradeCommodity(trade) {

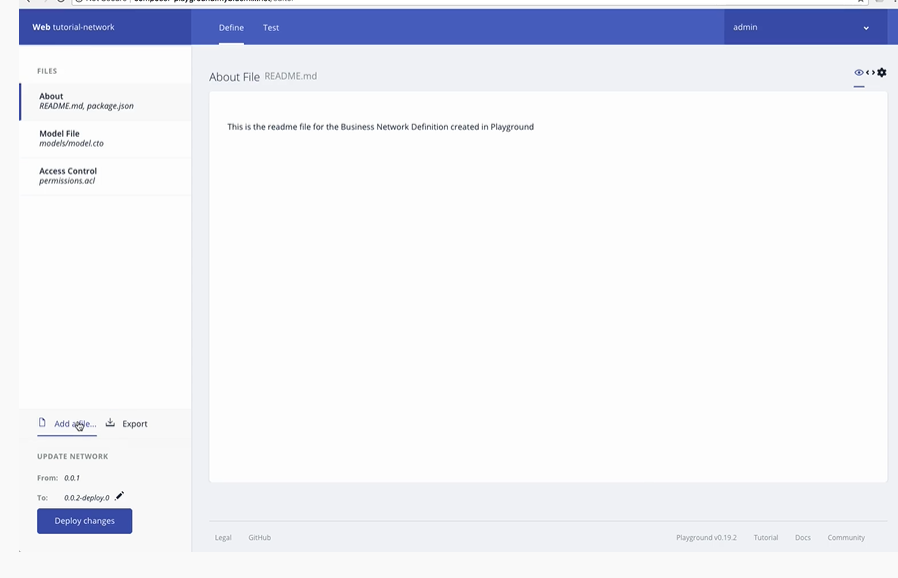
trade.commodity.owner = trade.newOwner;

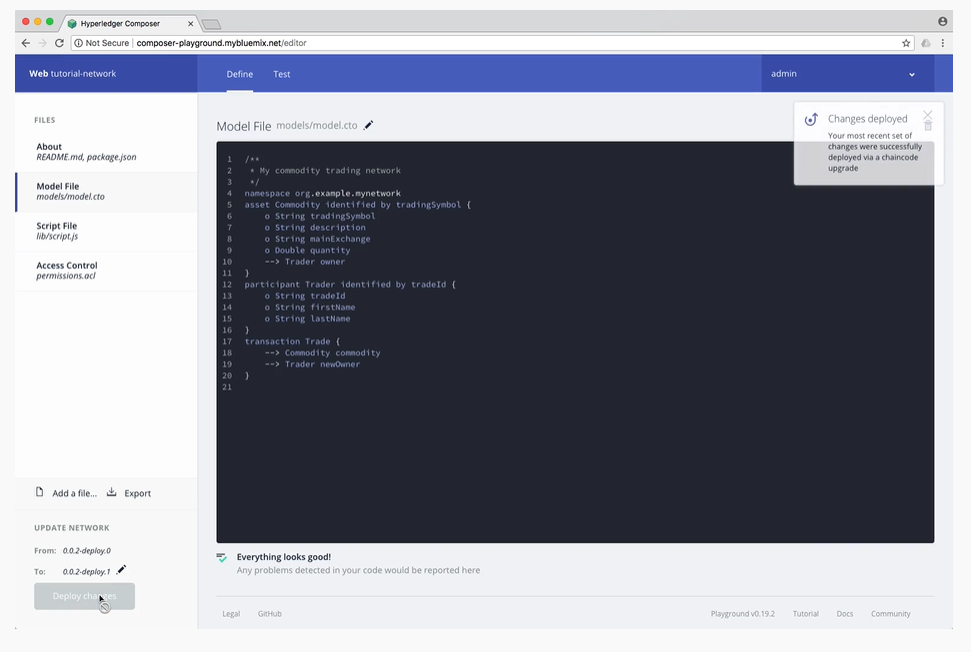
let assetRegistry = await getAssetRegistry('org.example.mynetwork.Commodity');

await assetRegistry.update(trade.commodity);

}

This function simply changes the owner property on a commodity based on the newOwner property on an incoming Trade transaction. It then persists the modified Commodity back into the asset registry, used to store Commodity instances.





The first thing we should add to our business network is two participants.

Ensure that you have the Trader tab selected on the left, and click Create New Participant in the upper right.

What you can see is the data structure of a Trader participant. We want some easily recognizable data, so delete the code that's there and paste the following:

Copy

{

"$class": "org.example.mynetwork.Trader",

"tradeId": "TRADER1",

"firstName": "Jenny",

"lastName": "Jones"

}

Click Create New to create the participant.

You should be able to see the new Trader participant you've created. We need another Trader to test our Trade transaction though, so create another Trader, but this time, use the following data:

Copy

{

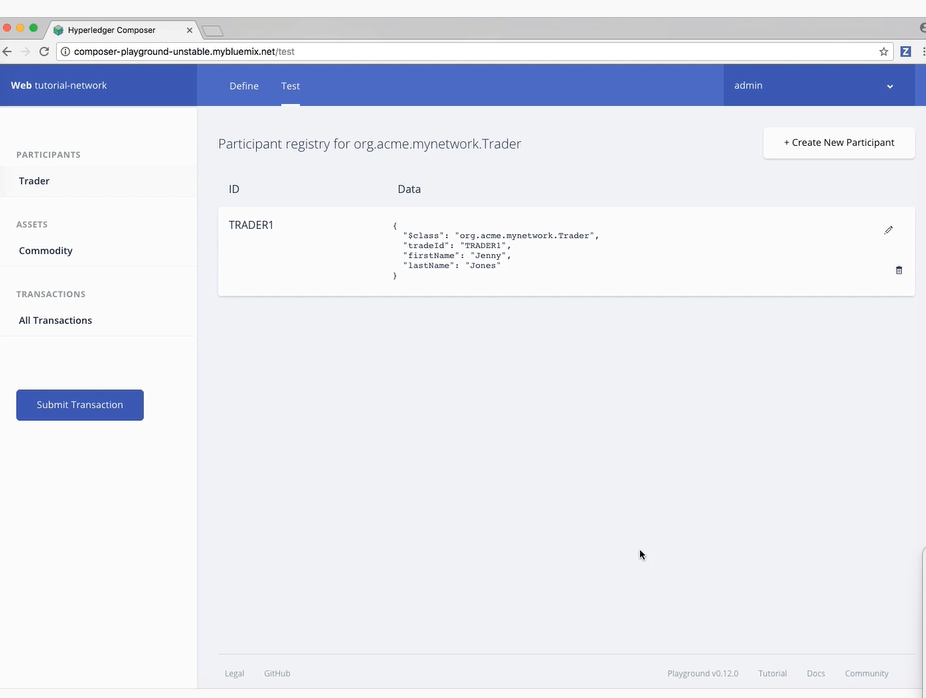
"$class": "org.example.mynetwork.Trader",

"tradeId": "TRADER2",

"firstName": "Amy",

"lastName": "Williams"

}



Now that we have two *Trader* participants, we need something for them to trade. Creating an asset is very similar to creating a participant. The *Commodity* we're creating will have an *owner* property indicating that it belongs to the *Trader* with the *tradeId* of TRADER1.

1. Click the **Commodity** tab under **Assets** and click **Create New Asset**.
2. Delete the asset data and replace it with the following:

Copy

{

"$class": "org.example.mynetwork.Commodity",

"tradingSymbol": "ABC",

"description": "Test commodity",

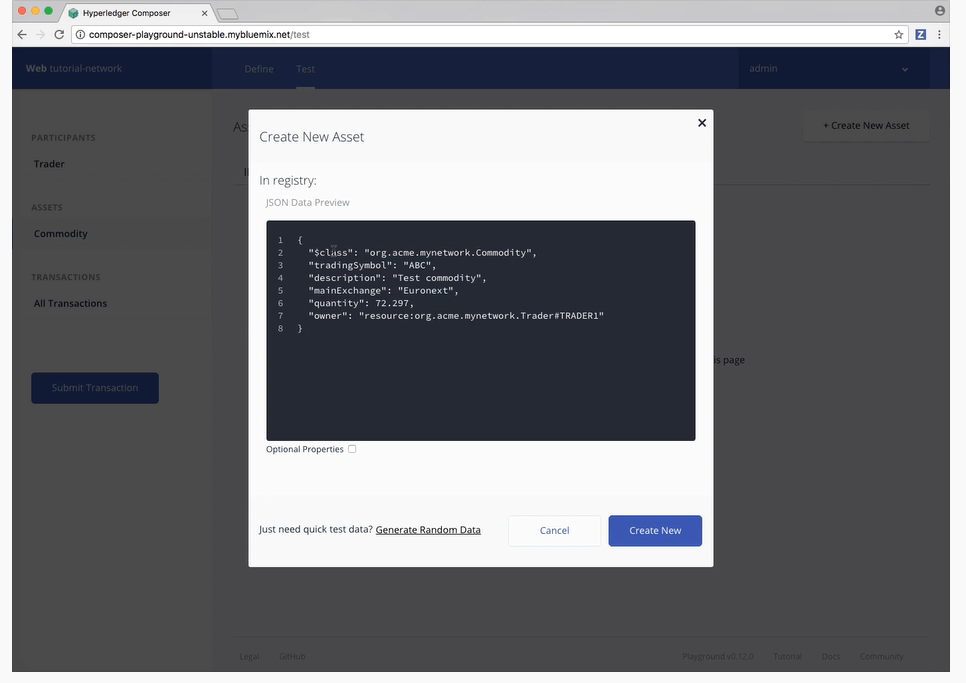
"mainExchange": "Euronext",

"quantity": 72.297,

"owner": "resource:org.example.mynetwork.Trader#TRADER1"

}

1. After creating this asset, you should be able to see it in the **Commodity** tab.

.To test the *Trade* transaction: Click the **Submit Transaction** button on the left.Ensure that the transaction type is *Trade*.Replace the transaction data with the following, or just change the details:

{

"$class": "org.example.mynetwork.Trade",

"commodity": "resource:org.example.mynetwork.Commodity#ABC",

"newOwner": "resource:org.example.mynetwork.Trader#TRADER2"

}

Click **Submit**.