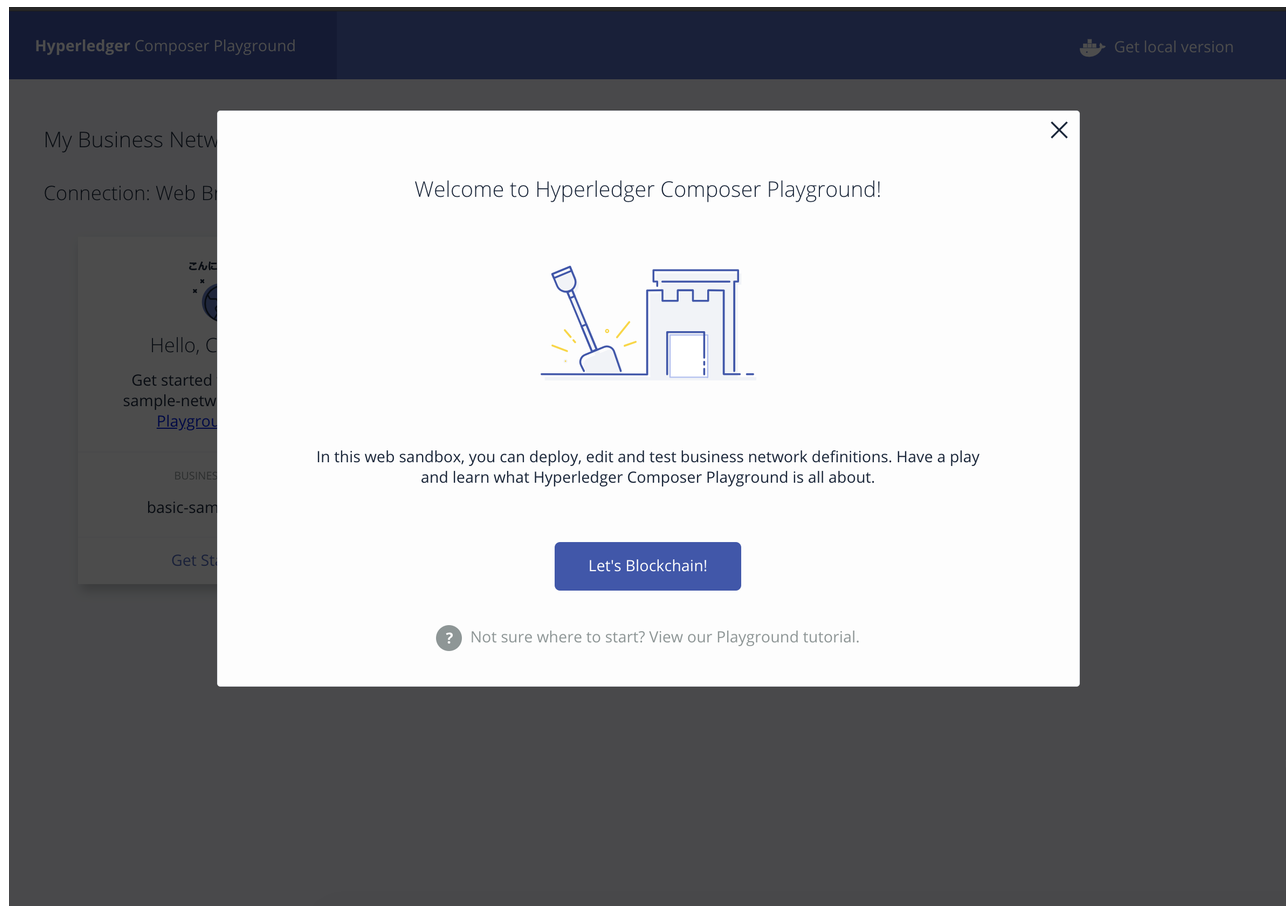


Lab

For the Lab we will create a network of a Meetup. This is not in any way a business use case. But today's lab will demonstrate the features of composer. It will all be done on the browser. That means no setup is necessary on your local machine.

<https://composer-playground.mybluemix.net/editor>


- Go to the URL Above.
- Click on Lets Blockchain.



- Click on Deploy New Business Network

My Business Networks

Connection: Web Browser




Hello, Composer!

Get started with the basic-sample-network, or view our [Playground tutorial](#)

BUSINESS NETWORK

basic-sample-network

Get Started →



Deploy a new business network

- Select `empty-business-network` and give the network a name. I am calling mine `meetup-network`
- Give the name of admin card. I am calling it `admin@meetup-network`
- Click Deploy

Hyperledger Composer Playground

Get local version

← My Wallet

ⓘ Not sure where to start? View our Playground tutorial.

Deploy New Business Network

1. BASIC INFORMATION

Give your new Business Network a name:

meetup-network

Describe what your Business Network will be used for:

Start from scratch with a blank business network

Give the network admin card that will be created a name

admin@meetup-network

2. MODEL NETWORK STARTER TEMPLATE

Choose a Business Network Definition to start with:

Choose a sample to play with, start a new project, or import your previous work

basic-sample-network

empty-business-network

Drop here to upload or [browse](#)

meetup-network

Start from scratch with a blank business network

CONNECTION PROFILE

BASED ON

empty-business-network

Start from scratch with a blank business network

Contains: 0 Participant Types, 0 Asset Types, and 0 Transaction Types

Deploy

Samples on npm

- Click Connect Now

Hyperledger Composer Playground

Get local version

My Business Networks

Connection: Web Browser

A

admin@meetup-network

USER ID

admin

BUSINESS NETWORK

meetup-network

Connect now →

Deploy a new business network

- Go to Model File Tab

The screenshot shows the 'Web meetup-network' interface. The top navigation bar includes 'Web meetup-network', 'Define', 'Test', and 'admin'. The left sidebar has a 'FILES' section with links to 'About', 'Model File', and 'Access Control'. The 'Model File' link is selected, showing the file 'models/model.cto'. The main content area displays the file's content, which is a CTO file with a license header and a namespace declaration. The status bar at the bottom indicates 'Everything looks good!'.

```

1  /*
2   * Licensed under the Apache License, Version 2.0 (the "License");
3   * you may not use this file except in compliance with the License.
4   * You may obtain a copy of the License at
5   *
6   * http://www.apache.org/licenses/LICENSE-2.0
7   *
8   * Unless required by applicable law or agreed to in writing, software
9   * distributed under the License is distributed on an "AS IS" BASIS,
10  * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
11  * See the License for the specific language governing permissions and
12  * limitations under the License.
13  */
14
15  namespace org.example.empty
16

```

Everything looks good!
Any problems detected in your code would be reported here

Legal GitHub Playground v0.20.1 Tutorial Docs Community

Here we will make changes.

In line 15 change the namespace to your namespace. I am calling it `org.meetup.ibmcode`. All other objects in the model file will be part of this namespace.

In our meetup we do two kinds of things. We have pizzas and we gain knowledge. These are our assets. We have two kinds of participants, audience and instructor. And we have two kinds of transactions, Eat and Learn.

Let's work on the eat transaction first.

First we create the Pizza asset. Add the following to model file.

```

asset Pizza identified by id {
    o String id
    o Integer amount
}

```

We just created an asset called pizza. It will be identified by its id. It has another field called amount.

Then we create the Audience Participant.

```
abstract participant Member identified by id {  
    o String id  
    o PresenceState state  
    --> Knowledge[] topics  
}  
  
participant Audience extends Member {  
    o FullState full  
}
```

Notice the `abstract` participant. This is because the instructor and audience share a super class member. Also you can see couple of unknown types such as `PresenceState` and `FullState`. These are couple of enums. Add them toward the top of the file.

```
enum PresenceState {  
    o PRESENT  
    o ABSENT  
}  
  
enum FullState {  
    o STARVING  
    o HUNGRY  
    o FULL  
}
```

We also need to add `Knowledge` asset.

```
asset Knowledge identified by id {
```

```

    o String id
    o String name
}

```

Finally we add the eat transaction.

```

transaction Eat {
    --> Audience member
    --> Pizza food
}

```

We see the `-->` symbol. This just denotes that `member` and `food` are relationships and not owned by the transactions.

Its now time to work on the logic of the transaction.

- Click on `Add a file...` in the left pane toward the bottom.

The screenshot shows the Web meetup-network IDE interface. The top bar includes the project name 'Web meetup-network', tabs for 'Define' and 'Test', and a user profile 'admin'. The left sidebar contains a 'FILES' panel with a tree view showing 'About' (README.md, package.json), 'Model File' (models/model.cto), and 'Access Control' (permissions.acl). Below the files panel are buttons for 'Add a file...' and 'Export', and a section for 'UPDATE NETWORK' with 'From: 0.0.1', 'To: 0.0.2-deploy.0', and a 'Deploy changes' button. The main editor area is titled 'Model File models/model.cto' and displays the following code:

```

1  /*
2  * Licensed under the Apache License, Version 2.0 (the "License");
3  * you may not use this file except in compliance with the License.
4  * You may obtain a copy of the License at
5  *
6  * http://www.apache.org/licenses/LICENSE-2.0
7  *
8  * Unless required by applicable law or agreed to in writing, software
9  * distributed under the License is distributed on an "AS IS" BASIS,
10 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
11 * See the License for the specific language governing permissions and
12 * limitations under the License.
13 */
14
15 namespace org.example.empty
16


```

Below the code editor, a green checkmark icon and the text 'Everything looks good!' are displayed, followed by the message 'Any problems detected in your code would be reported here'. The bottom of the interface features a footer with links for 'Legal', 'GitHub', 'Playground v0.20.1', 'Tutorial', 'Docs', and 'Community'.

- Select `Script File (.js)`

Add a file

Upload a file from your computer...

 Drop here to upload or [browse](#)

☐ **Model File (.cto)**
Define Assets, Participants and Transactions using Hyperledger Composer modelling language.

☐ **Script File (.js)**
Define the logic of transaction executions using JavaScript.

☐ **Query File (.qry)**
Define the queries in here (Note: you can only have 1 of these per .bna).

☐ **Access Control File (permissions.acl)**
Define your access controls here (Note: you can only have 1 of these per .bna).

Cancel

Add

- Now we can edit this file.

The screenshot shows a web application interface for 'Web meetup-network'. At the top, there's a navigation bar with 'Define' and 'Test' tabs, and a user profile 'admin'. On the left, a sidebar lists files: 'About' (README.md, package.json), 'Model File' (models/model.cto), 'Script File' (lib/script.js - selected), and 'Access Control' (permissions.acl). Below the sidebar, there are buttons for 'Add a file...' and 'Export', and a section for 'UPDATE NETWORK' with a 'Deploy changes' button. The main area shows the 'Script File' editor for 'lib/script.js'. The code in the editor is:

```
1 /**
2  * New script file
3  */
```

 Below the code, a green checkmark icon and the text 'Everything looks good!' are displayed, followed by a note: 'Any problems detected in your code would be reported here'. At the bottom of the interface, there are links for 'Legal', 'GitHub', 'Playground v0.20.1', 'Tutorial', 'Docs', and 'Community'.

Add the following in script.js file.

```
/**
 * @param {org.meetup.ibmcode.Eat} eat - the eat transaction
 * @transaction
 */
```

This format of comment needs to be there before any transaction function. This tells the Js function which resource it gets passed when it is called. Notice the namespace name. It should match your namespace name.

Right after the comments above add the function. The name of the function or the variable passed does not matter. We could call it whatever we want. But they will connect to our model via the `@param` tag.

```
async function eat(eat) {
  let member = eat.member;

  let willEat = 0;
```



```
if(member.state !== 'PRESENT') {
  throw new Error ('Not Present Members can not eat');
}

if (member.full === 'STARVING') {
  willEat = 2;
} else if(member.full === 'HUNGRY') {
  willEat = 1;
} else {
  willEat = 0;
}

let food = eat.food;
if (food.amount < willEat) {
  throw new Error ('Not Enough Food');
}

food.amount = food.amount - willEat;
member.full = 'FULL';

const foodRegistry = await getAssetRegistry('org.meetup.ibmcode.Pizza');
await foodRegistry.update(food);

const memberRegistry = await getParticipantRegistry('org.meetup.ibmcode.Audience');
await memberRegistry.update(member);
}
```

In the code all we are doing is checking if our audience can eat some pizza. There is logic to check if they are present to the meetup also if there is enough food. After the transaction the member should be full and number of pizzas left should decrease. We update it in the end using `getAssetRegistry` and `getParticipantRegistry`

Now that everything looks good. (No errors)

- Click `Deploy Changes`

The screenshot shows the Web meetup-network IDE interface. The top bar has tabs for 'Web meetup-network', 'Define', 'Test', and 'admin'. The left sidebar contains a 'FILES' section with 'About' (README.md, package.json), 'Model File' (models/model.cto), 'Script File' (lib/script.js), and 'Access Control' (permissions.acl). Below this is an 'UPDATE NETWORK' section showing a version update from 0.0.1 to 0.0.2-deploy.0 with a 'Deploy changes' button. The main area displays the 'Model File models/model.cto' with the following code:

```
33 }
34
35 asset Knowledge identified by id {
36   o String id
37   o String name
38 }
39
40
41 abstract participant Member identified by id {
42   o String id
43   o PresenceState state
44   --> Knowledge[] topics
45 }
46
47 participant Audience extends Member {
48   o FullState full
49 }
50
51
52 transaction Eat {
53   --> Audience member
54   --> Pizza food
55 }
```

Below the code editor, a green checkmark icon and the text 'Everything looks good!' are displayed, followed by the message 'Any problems detected in your code would be reported here'. The bottom of the interface includes links for 'Legal', 'GitHub', 'Playground v0.20.1', 'Tutorial', 'Docs', and 'Community'.

- Go to `Test` tab. You should see the following screen.

Web meetup-network

DefineTest

admin

PARTICIPANTS

Audience

ASSETS

Knowledge

Pizza


TRANSACTIONS

All Transactions

Submit Transaction

Participant registry for org.meetup.ibmcode.Audience

+ Create New Participant

ID	Data
 <p>This registry is empty!</p> <p>To create resources in this registry click create new at the top of this page</p>	

[Legal](#) [GitHub](#) [Playground v0.20.1](#) [Tutorial](#) [Docs](#) [Community](#)

- Create a new participant

```
{
  "$class": "org.meetup.ibmcode.Audience",
  "full": "STARVING",
  "id": "1",
  "state": "PRESENT",
  "topics": []
}
```

×

Create New Participant

In registry: **org.meetup.ibmcode.Audience**

JSON Data Preview

```
1  {
2    "$class": "org.meetup.ibmcode.Audience",
3    "full": "STARVING",
4    "id": "1",
5    "state": "PRESENT",
6    "topics": []
7  }
```

☐ Optional Properties

Just need quick test data? [Generate Random Data](#)

Cancel

Create New

- Create a Pizza Asset

```
{
  "$class": "org.meetup.ibmcode.Pizza",
  "id": "1",
  "amount": 0
}
```

- Click on submit transaction.

Web meetup-network

DefineTest

PARTICIPANTS

Audience

ASSETS

Knowledge

Pizza

TRANSACTIONS

All Transactions

Submit Transaction

Asset registry for org.meetup.ibmcc

ID

1

- Submit the transaction. Member is set to audience id#1 and pizza #1

Submit Transaction

Transaction TypeEat

JSON Data Preview

```
1  {
2    "$class": "org.meetup.ibmcode.Eat",
3    "member": "resource:org.meetup.ibmcode.Audience#1",
4    "food": "resource:org.meetup.ibmcode.Pizza#1"
5  }
```

☐ Optional Properties

Just need quick test data? [Generate Random Data](#)

Cancel

Submit

- You should see an error. (Not Enough Food)

Submit Transaction

Transaction TypeEat

JSON Data Preview

```
1  {
2    "$class": "org.meetup.ibmcode.Eat",
3    "member": "resource:org.meetup.ibmcode.Audience#1",
4    "food": "resource:org.meetup.ibmcode.Pizza#1"
5  }
```

☐ Optional Properties

Error: Not Enough Food

Just need quick test data? [Generate Random Data](#)

Cancel

Submit

- Lets go update our pizza asset. Click on the pen on the asset.

Web meetup-network

DefineTest

admin

PARTICIPANTS

Audience

ASSETS

Knowledge

Pizza

TRANSACTIONS

All Transactions

Submit Transaction

Asset registry for org.meetup.ibmcode.Pizza

+ Create New Asset

ID	Data
1	<div><pre>{ "\$class": "org.meetup.ibmcode.Pizza", "id": "1", "amount": 0 }</pre></div>

- Update the amount to 5 (Or any number really)

ID	Data
1	<div><pre>{ "\$class": "org.meetup.ibmcode.Pizza", "id": "1", "amount": 5 }</pre></div>

- Go Back and submit the transaction again.
- It was a success this time. We can see our pizza asset updated as per our logic.

ID	Data
1	<div><pre>{ "\$class": "org.meetup.ibmcode.Pizza", "id": "1", "amount": 3 }</pre></div>

- Go to All transactions.
- You can see all our changes logged.

PARTICIPANTS				
Audience				
ASSETS				
Knowledge	2018-09-30, 22:32:45	Eat	admin (NetworkAdmin)	view record
Pizza	2018-09-30, 22:31:43	UpdateAsset	admin (NetworkAdmin)	view record
TRANSACTIONS				
All Transactions	2018-09-30, 22:25:11	AddAsset	admin (NetworkAdmin)	view record
	2018-09-30, 22:24:34	AddParticipant	admin (NetworkAdmin)	view record

- Our Update is also a transaction. This is the ledger and is immutable.

We can add another transaction to this. Complete code is attached in the repo. Look at lib/script.js and model/model.cto for the complete code.