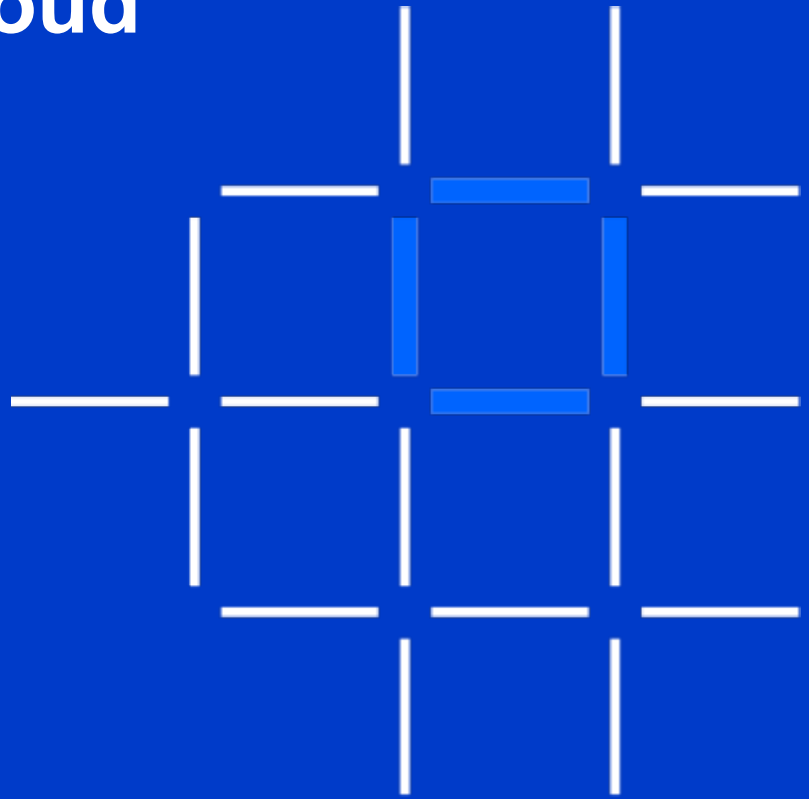
Create and Deploy a Blockchain App in the Cloud

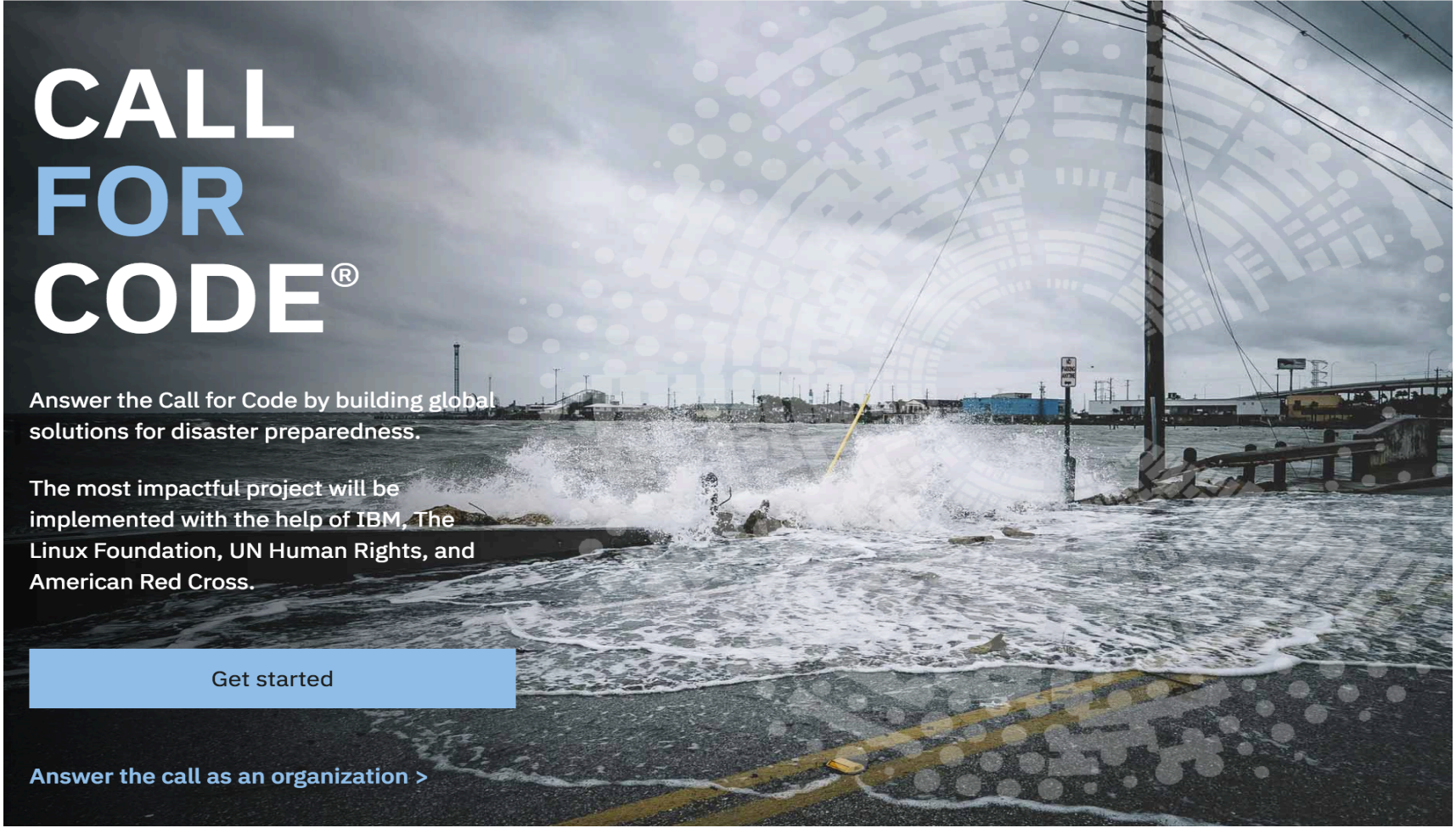


Lennart Frantzell,
IBM Developer Advocate,
San Francisco
alf@us.ibm.com

V1 25 June 2018

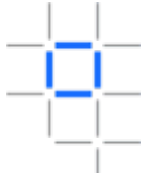
IBM **Blockchain**

IBM



IBM **Blockchain** https://developer.ibm.com/callforcode/resources/financial-networks/

How did it all start?

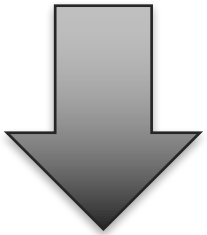


October 2008. It all started with Satoshi Nakamoto and his paper Bitcoin: A Peer-to-Peer Electronic Cash System which addressed a key problem in electronic commerce:

> *A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution.*
>
> *Digital signatures provide part of the solution, but the main benefits are lost if a trusted third party is still required to prevent double-spending.*
>
> ***We propose a solution to the double-spending problem using a peer-to-peer network.***

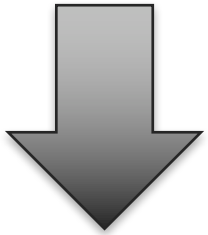
https://bitcoin.org/bitcoin.pdf

A View from 50 000 feet

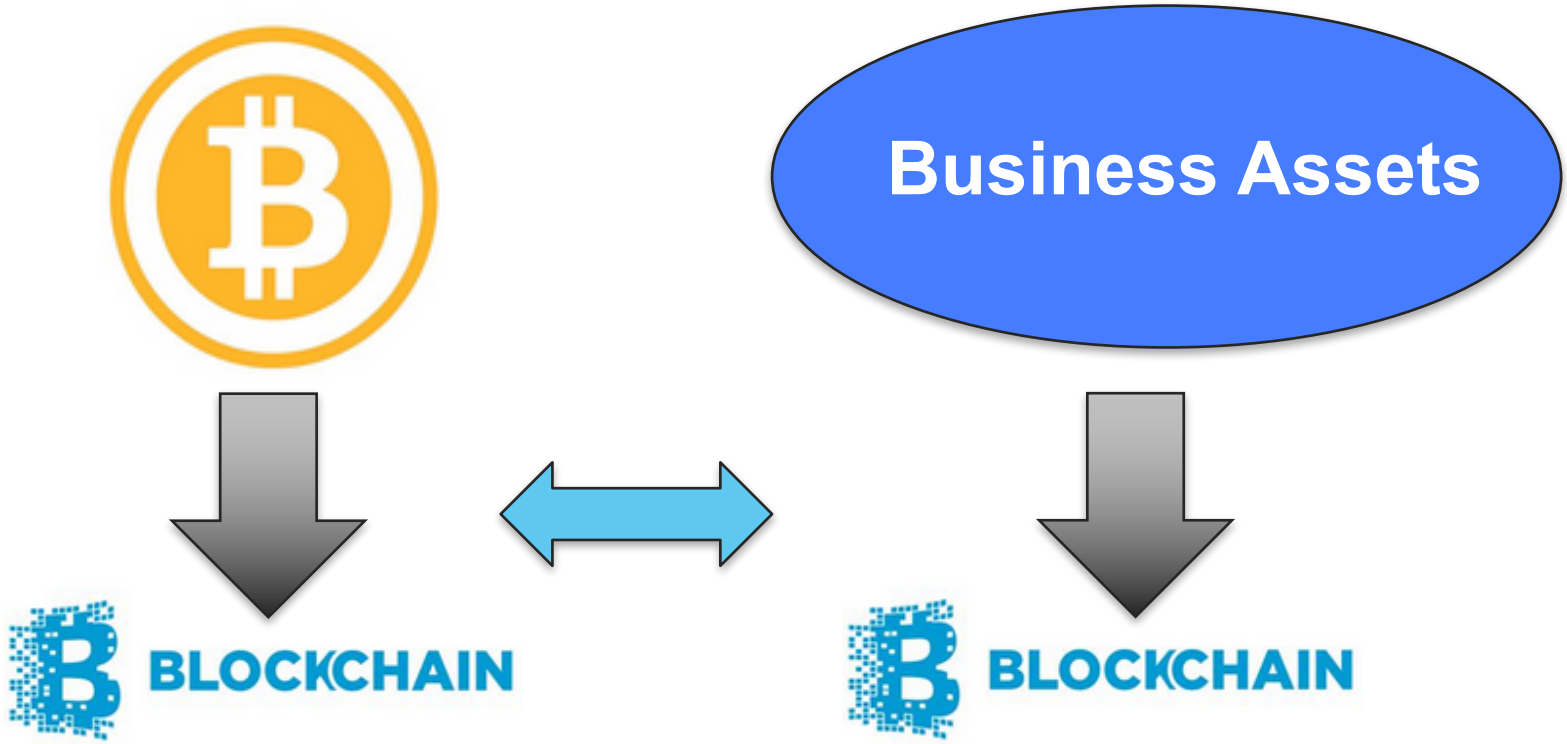


VS

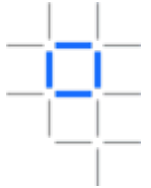
Business Assets



A View from 50 000 feet



Bitcoin vs Blockchain



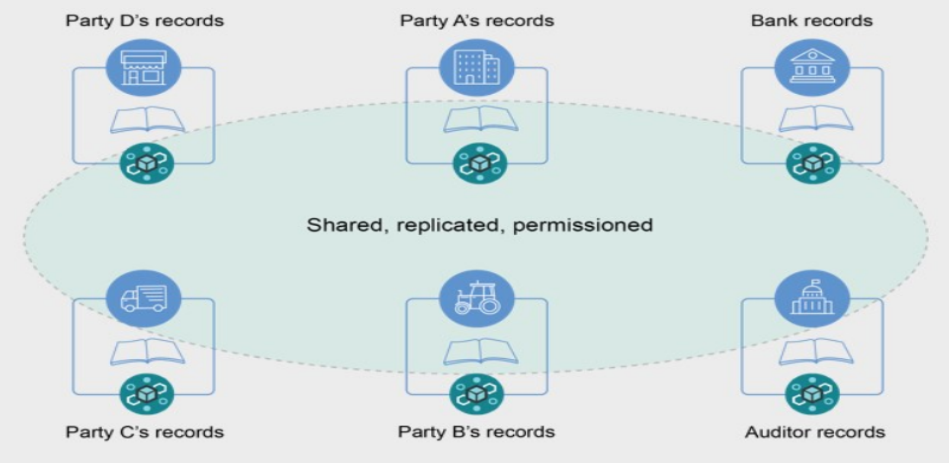
Bitcoin	Blockchain
Cryptocurrency	Assets
Anonymity	Identity
Proof of work (mining)	Selective endorsement (consensus)

The Blockchain, a shared distributed ledger

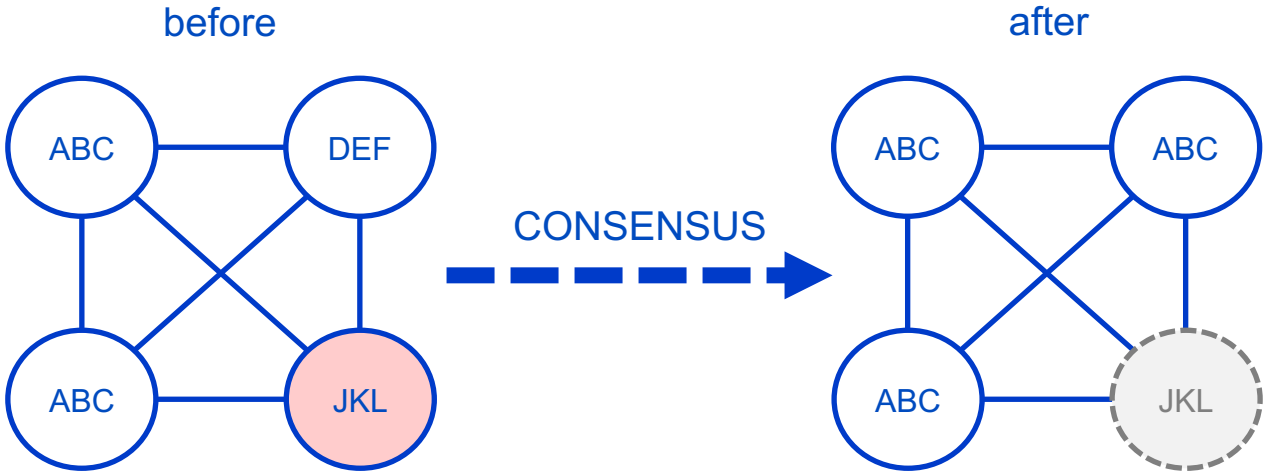
1) A distributed ledger is a type of database that is shared, replicated, and synchronized among the members of a network. **The distributed ledger records the transactions, such as the exchange of assets or data, among the participants in the network.**

2) Every record in the distributed ledger has a timestamp and unique cryptographic signature, **making the ledger immutable** and providing an auditable history of all transactions in the network.

3) Participants in the network govern and agree **by consensus** on the updates to the records in the ledger. **No central, third-party mediator, such as a financial institution or clearinghouse, is involved.** **No reconciliation**



Consensus: The process of maintaining a consistent ledger



Keep all peers up-to-date
Fix any peers in error
Ignoring all malicious nodes

Consensus algorithms have different strengths and weaknesses

Solo / No-ops

Validators apply received transactions without consensus
PROs: Very quick; suited to development
CONS: No consensus; can lead to divergent chains
Example usage: Hyperledger Fabric V1

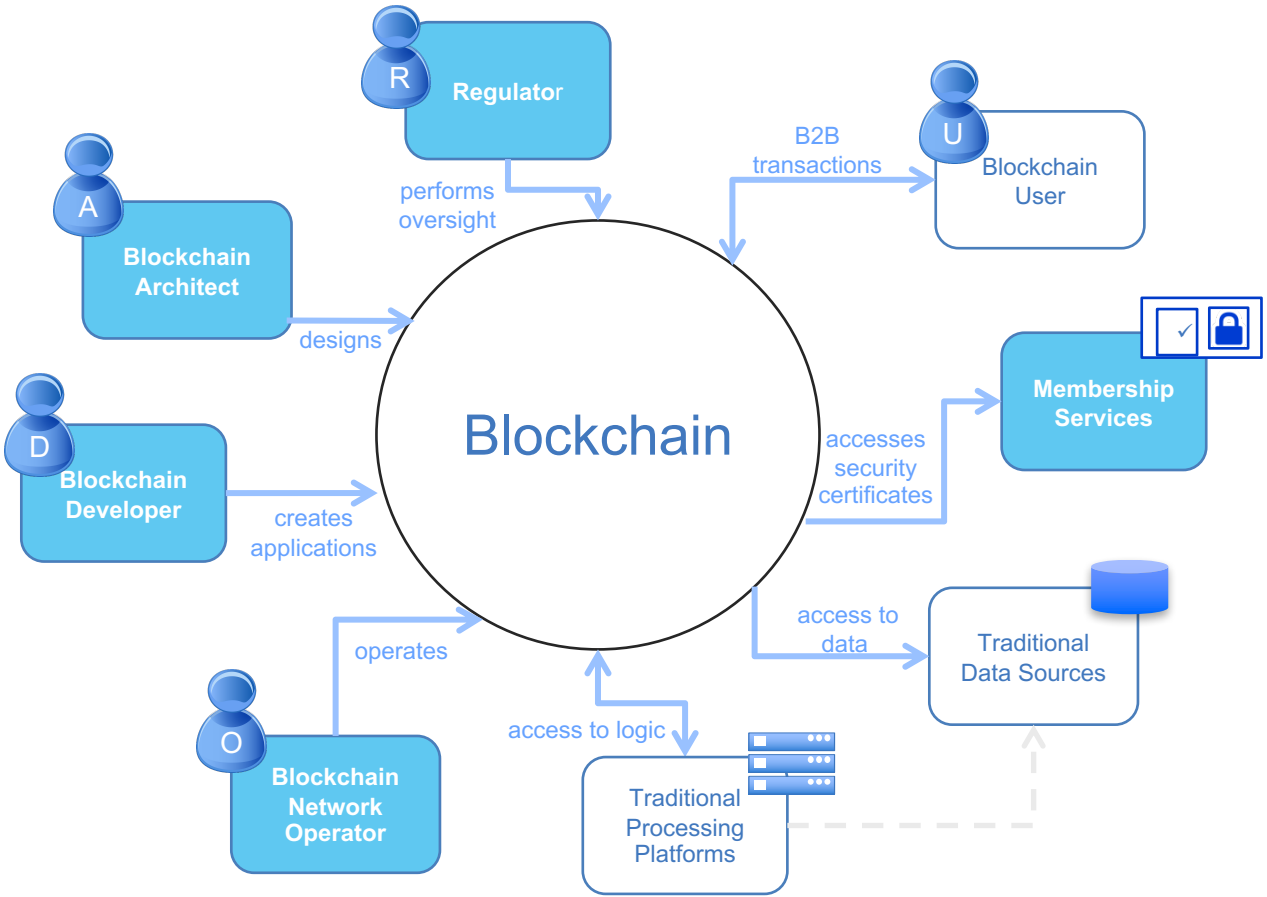
PBFT-based

Practical Byzantine Fault Tolerance implementations
PROs: Reasonably efficient and tolerant against malicious peers
CONS: Validators are known and totally connected
Example usage: Hyperledger Fabric V0.6

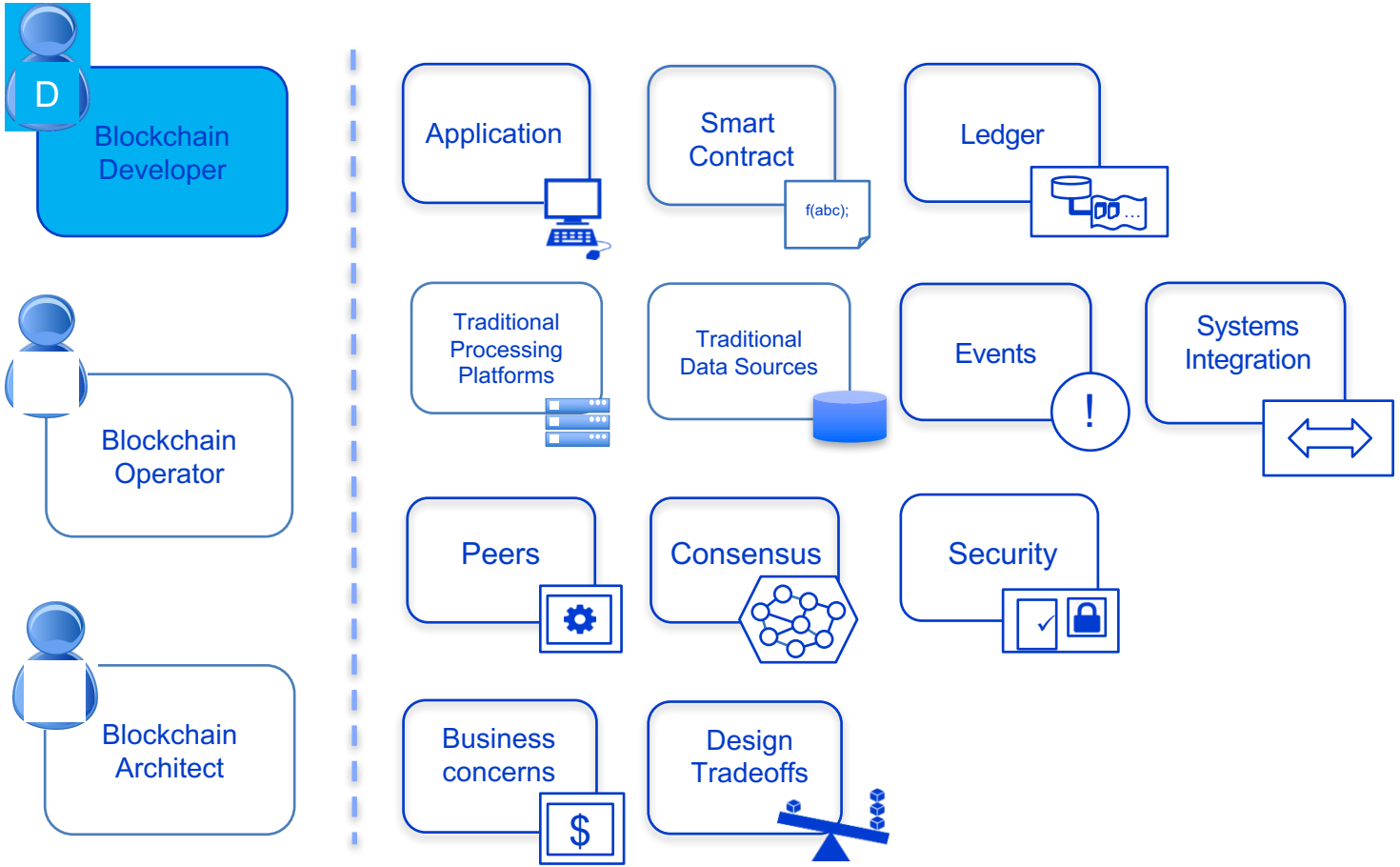
Kafka / Zookeeper

Ordering service distributes blocks to peers
PROs: Efficient and fault tolerant
CONS: Does not guard against malicious activity
Example usage: Hyperledger Fabric V1

Actors in a Blockchain solution



Summary of Key Concepts



Blockchain Use Cases

Supply chain, asset registration, identity services, fraud prevention and compliance

IBM Blockchain government point of view

PDF See the infographic (50.6KB)

Payment and digital currency

IBM's universal blockchain payments solution

PDF See the infographic (1.4MB)

Payment and digital currency

IBM announces major blockchain solution to speed up global payments

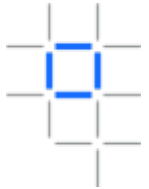
→ Read the press release

European trade finance network: we.trade

Spreading opportunity to small and medium sized businesses in traditionally underserved markets.

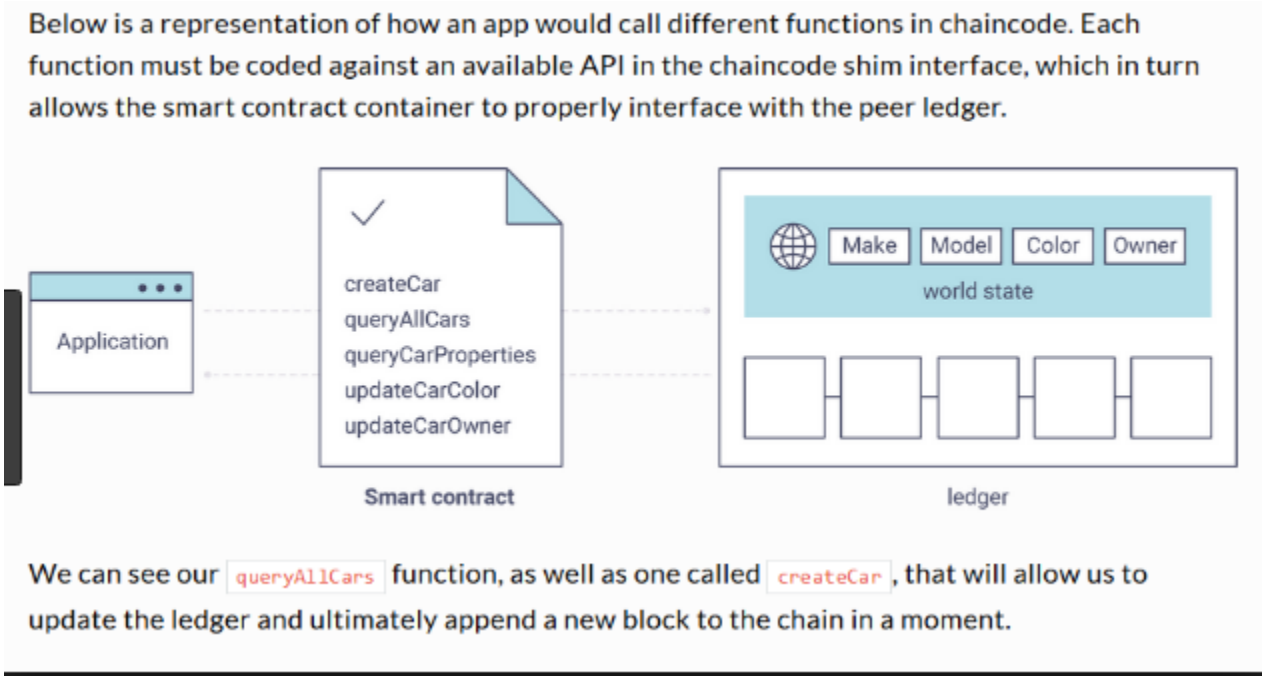
▷ Watch the video (02:48)

https://www.ibm.com/blockchain/use-cases/



How do we program the ledger?

With Chaincode, aka Smart Contracts



Like Stored Procedures in databases

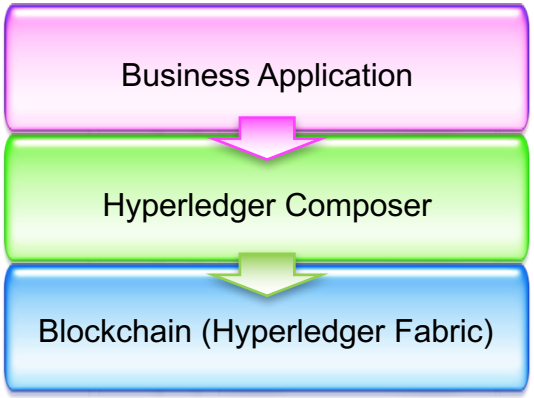
https://hyperledger-fabric.readthedocs.io/en/release-1.1/

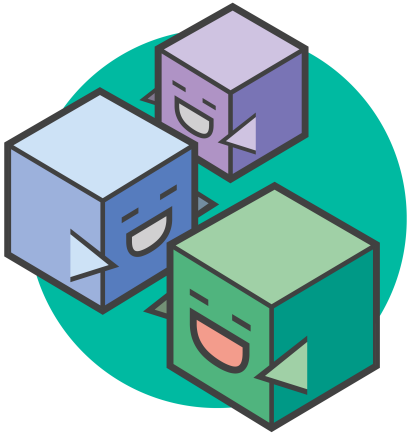
https://composer-playground.mybluemix.net/

Hyperledger Composer: Accelerating time to value

https://hyperledger.github.io/composer/

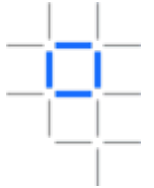
- A suite of high level application **abstractions** for business networks
- Emphasis on business-centric vocabulary for quick solution creation
- Reduce risk, and increase understanding and flexibility

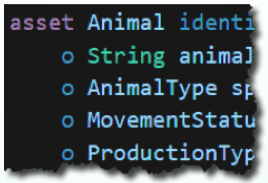




- Features
  - Model your business networks, test and expose via APIs
  - Applications invoke APIs transactions to interact with business network
  - Integrate existing systems of record using loopback/REST
- Fully open and part of Linux Foundation Hyperledger
- Try it in your web browser now: http://composer-playground.mybluemix.net/

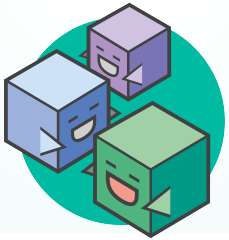
Extensive, Familiar, Open Development Toolset





Data modelling

JavaScript business logic



Web playground

Client libraries

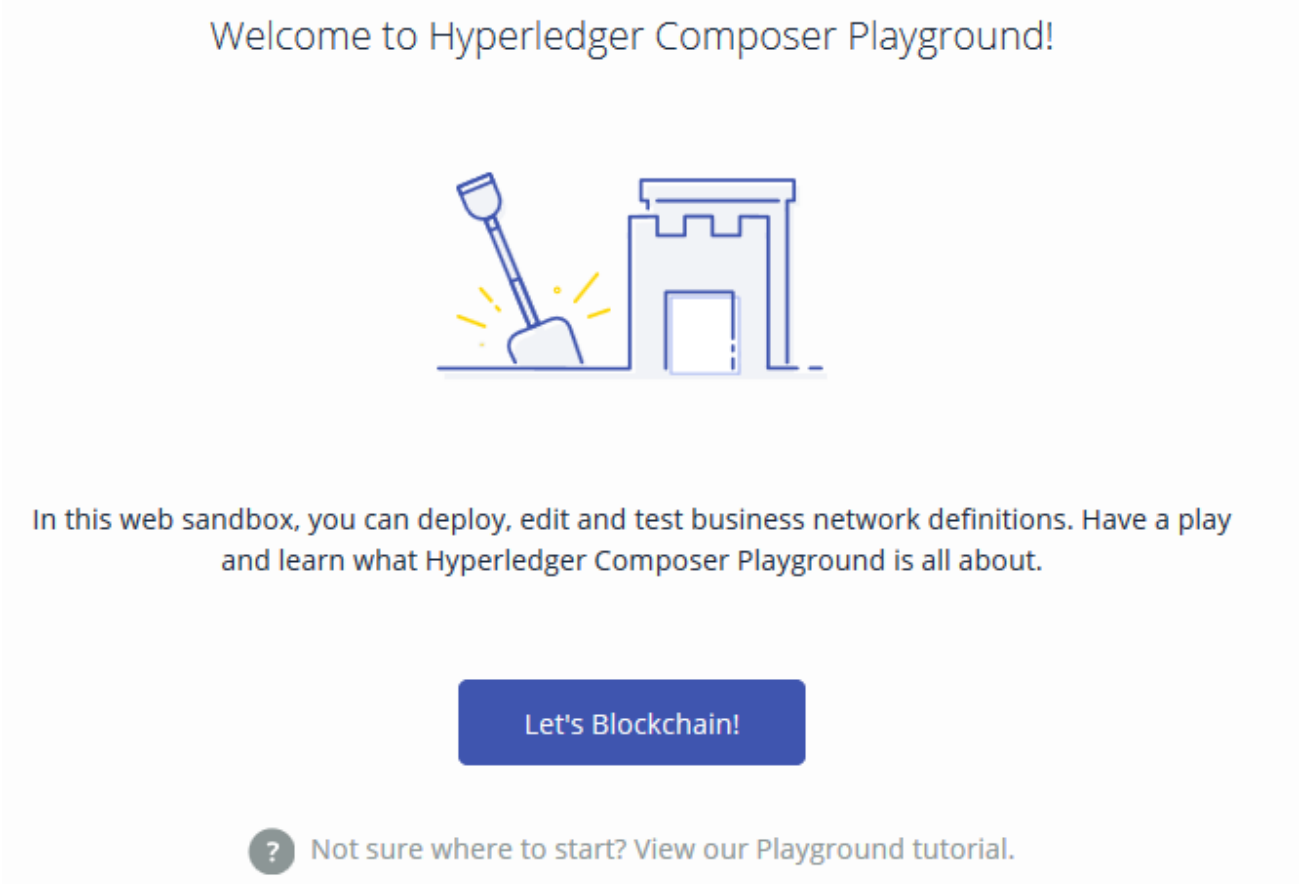
Editor support

CLI utilities

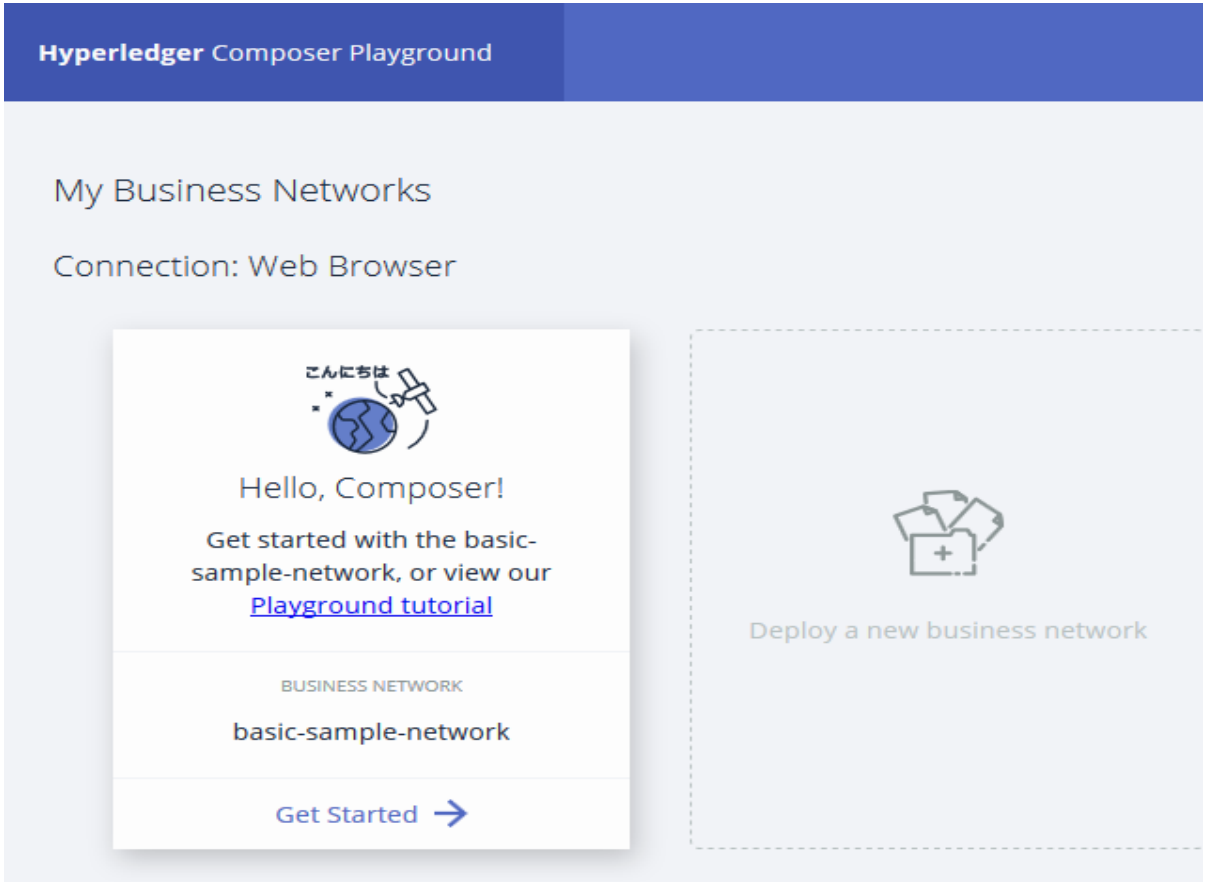
Code generation

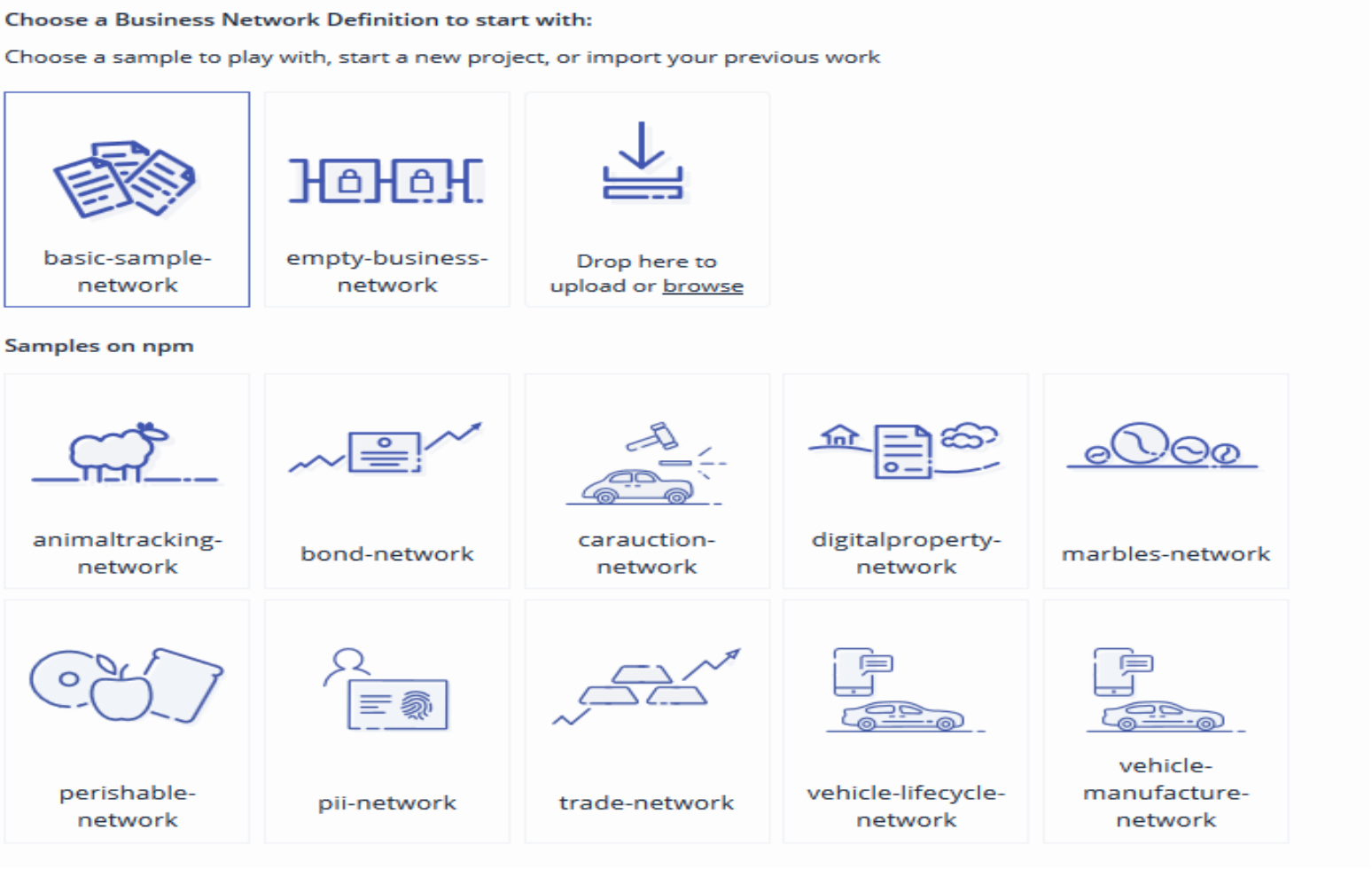
Existing systems and data

https://composer-playground.mybluemix.net/



https://hyperledger.github.io/composer/latest/tutorials/playground-tutorial


Hyperledger Composer Playground
My Business Networks
Connection: Web Browser
こんにちは
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


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
Samples on npm
animaltracking-network
bond-network
carauction-network
digitalproperty-network
marbles-network
perishable-network
pii-network
trade-network
vehicle-lifecycle-network
vehicle-manufacture-network

FILES

About
README.md, package.json

Model File
models/sample.cto

Script File
lib/sample.js

Access Control
permissions.acl

Add a file... Export

UPDATE NETWORK

From: *0.2.6-20180530153450*

To: *0.2.6-deploy.0*

Deploy changes

About File README.md

Basic Sample Business Network

> *This is the "Hello World" of Hyperledger Composer samples, which demonstrates the core functionality of Hyperledger Composer by changing the value of an asset.*

This business network defines:

Participant `SampleParticipant`

Asset `SampleAsset`

Transaction `SampleTransaction`

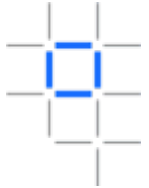
Event `SampleEvent`

SampleAssets are owned by a SampleParticipant, and the value property on a SampleAsset can be modified by submitting a SampleTransaction. The SampleTransaction emits a SampleEvent that notifies applications of the old and new values for each modified SampleAsset.

To test this Business Network Definition in the **Test** tab:

Model File models/sample.cto

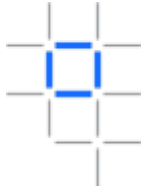
```
 */

/**
 * Sample business network definition.
 */
namespace org.example.basic

asset SampleAsset identified by assetId {
  o String assetId
  --> SampleParticipant owner
  o String value
}

participant SampleParticipant identified by participantId {
  o String participantId
  o String firstName
  o String lastName
}

transaction SampleTransaction {
  --> SampleAsset asset
  o String newValue
}

event SampleEvent {
  --> SampleAsset asset
  o String oldValue
  o String newValue
}

```



Script File

```
 */

/* global getAssetRegistry getFactory emit */

/**
 * Sample transaction processor function.
 * @param {org.example.basic.SampleTransaction} tx The sample transaction instance.
 * @transaction
 */
async function sampleTransaction(tx) {  // eslint-disable-line no-unused-vars

    // Save the old value of the asset.
    const oldValue = tx.asset.value;

    // Update the asset with the new value.
    tx.asset.value = tx.newValue;

    // Get the asset registry for the asset.
    const assetRegistry = await getAssetRegistry('org.example.basic.SampleAsset');
    // Update the asset in the asset registry.
    await assetRegistry.update(tx.asset);

    // Emit an event for the modified asset.
    let event = getFactory().newEvent('org.example.basic', 'SampleEvent');
    event.asset = tx.asset;
    event.oldValue = oldValue;
    event.newValue = tx.newValue;
    emit(event);
}
```

```
 * Sample access control list.
 */
rule EverybodyCanReadEverything {
    description: "Allow all participants read access to all resources"
    participant: "org.example.basic.SampleParticipant"
    operation: READ
    resource: "org.example.basic.*"
    action: ALLOW
}

rule EverybodyCanSubmitTransactions {
    description: "Allow all participants to submit transactions"
    participant: "org.example.basic.SampleParticipant"
    operation: CREATE
    resource: "org.example.basic.SampleTransaction"
    action: ALLOW
}

rule OwnerHasFullAccessToTheirAssets {
    description: "Allow all participants full access to their assets"
    participant(p): "org.example.basic.SampleParticipant"
    operation: ALL
    resource(r): "org.example.basic.SampleAsset"
    condition: (r.owner.getIdentifier() === p.getIdentifier())
    action: ALLOW
}
```



```
rule SystemACL {
    description: "System ACL to permit all access"
    participant: "org.hyperledger.composer.system.Participant"
    operation: ALL
    resource: "org.hyperledger.composer.system.**"
    action: ALLOW
}

rule NetworkAdminUser {
    description: "Grant business network administrators full access to user resources"
    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
}
```

FILES

About
README.md, package.json

Model File
models/sample.cto

Script File
lib/sample.js

Access Control
permissions.acl

Add a file... Export

UPDATE NETWORK

From: *0.2.6-20180530153450*

To: *0.2.6-deploy.0*

Deploy changes

About File README.md

Basic Sample Business Network

> *This is the "Hello World" of Hyperledger Composer samples, which demonstrates the core functionality of Hyperledger Composer by changing the value of an asset.*

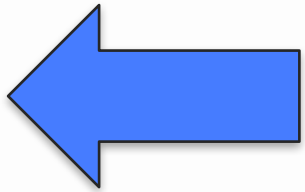
This business network defines:

Participant `SampleParticipant`

Asset `SampleAsset`

Transaction `SampleTransaction`

Event `SampleEvent`



SampleAssets are owned by a SampleParticipant, and the value property on a SampleAsset can be modified by submitting a SampleTransaction. The SampleTransaction emits a SampleEvent that notifies applications of the old and new values for each modified SampleAsset.

To test this Business Network Definition in the **Test** tab:

Create a `SampleParticipant` participant:

```
{
  "$class": "org.example.basic.SampleParticipant",
  "participantId": "Toby",
  "firstName": "Tobias",
  "lastName": "Hunter"
}
```

Create a `SampleAsset` asset:

```
{
  "$class": "org.example.basic.SampleAsset",
  "assetId": "assetId:1",
  "owner": "resource:org.example.basic.SampleParticipant#Toby",
  "value": "original value"
}
```

Submit a `SampleTransaction` transaction:

```
{
  "$class": "org.example.basic.SampleTransaction",
  "asset": "resource:org.example.basic.SampleAsset#assetId:1",
  "newValue": "new value"
}
```

After submitting this transaction, you should now see the transaction in the Transaction Registry and that a `SampleEvent` has been emitted. As a result, the value of the `assetId:1` should now be `new value` in the Asset Registry.

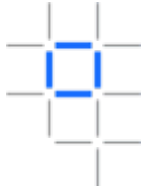

Define
Test
admin
Participant registry for org.example.basic.SampleParticipant
+ Create New Participant
ID
Data
Toby
{
"$class": "org.example.basic.SampleParticipant",
"participantId": "Toby",
"firstName": "Tobias",
"lastName": "Hunter"
}

Asset registry for org.example.basic.SampleAsset

+ Create New Asset

ID	Data

assetId:1	{
	"$class": "org.example.basic.SampleAsset",
	"assetId": "assetId:1",
	"owner": "resource:org.example.basic.SampleParticipant#Toby",
	"value": "original value"
	}



Submit Transaction

Transaction Type SampleTransaction

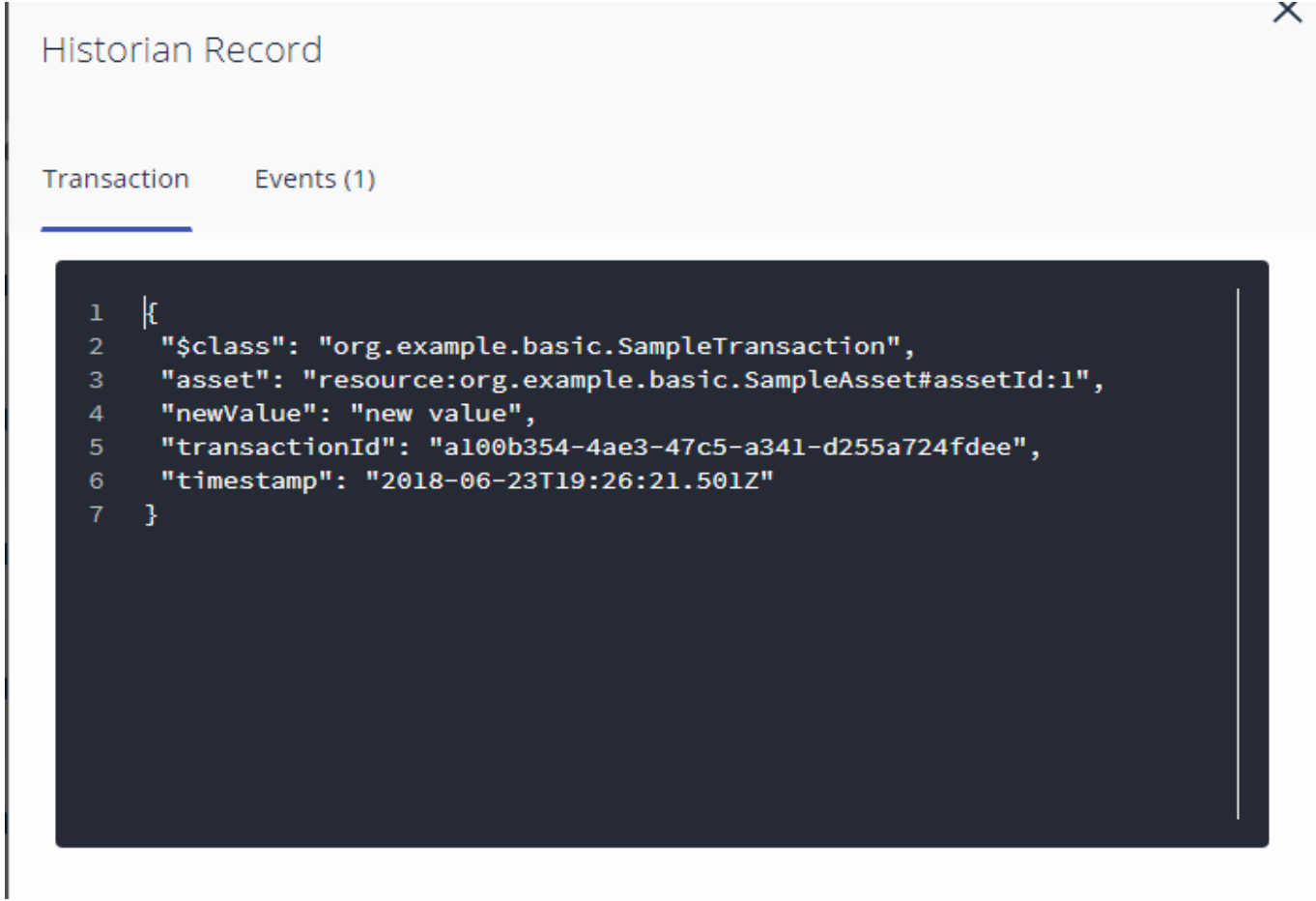
JSON Data Preview

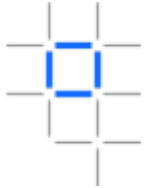
```
{
  "$class": "org.example.basic.SampleTransaction",
  "asset": "resource:org.example.basic.SampleAsset#assetId:1",
  "newValue": "new value"
}
```

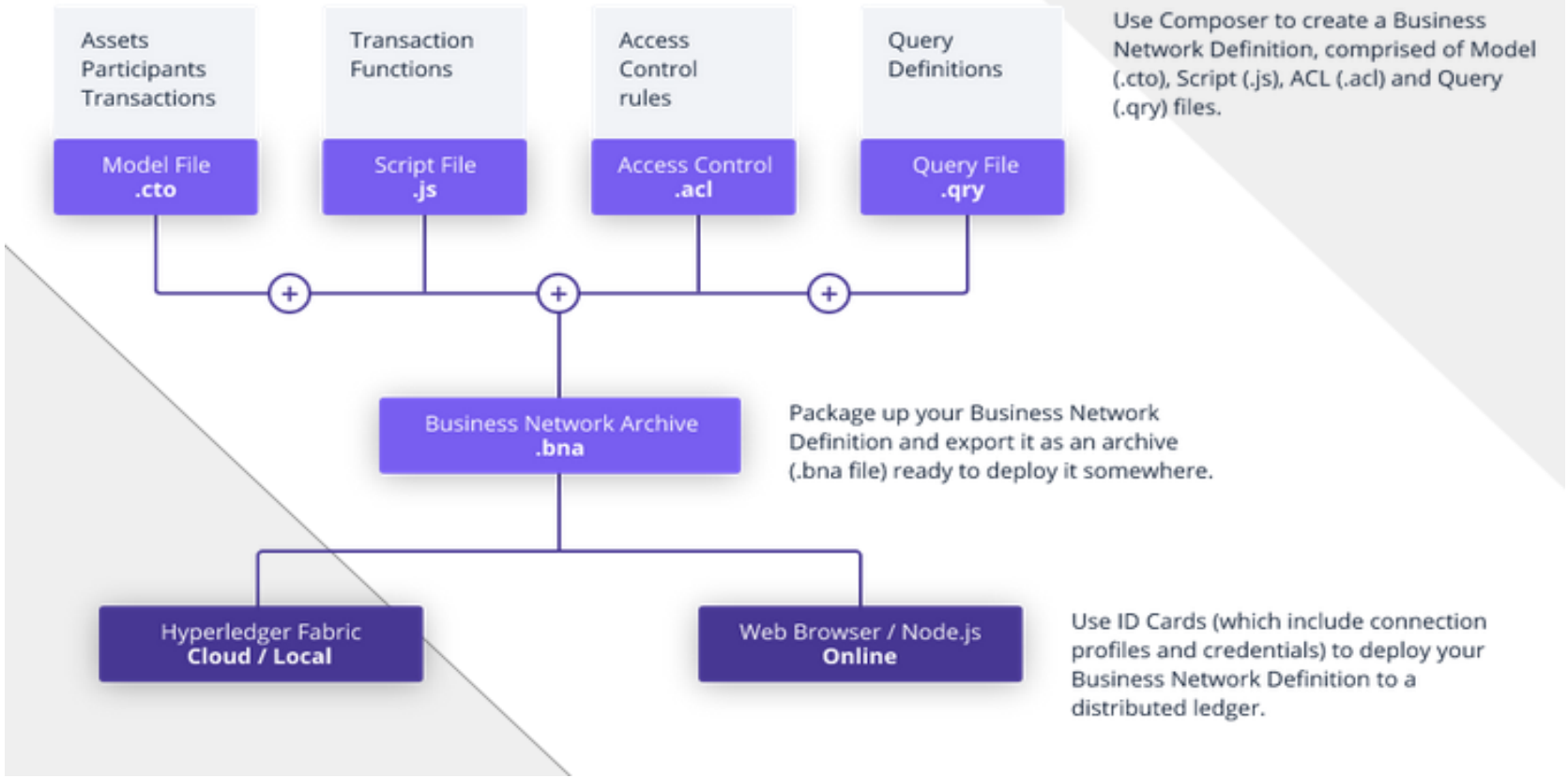
Optional Properties

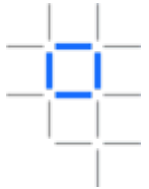
Just need quick test data? Generate Random Data

Cancel Submit


Historian Record
Transaction
Events (1)
{
"$class": "org.example.basic.SampleTransaction",
"asset": "resource:org.example.basic.SampleAsset#assetId:1",
"newValue": "new value",
"transactionId": "a100b354-4ae3-47c5-a341-d255a724fdee",
"timestamp": "2018-06-23T19:26:21.501Z"
}

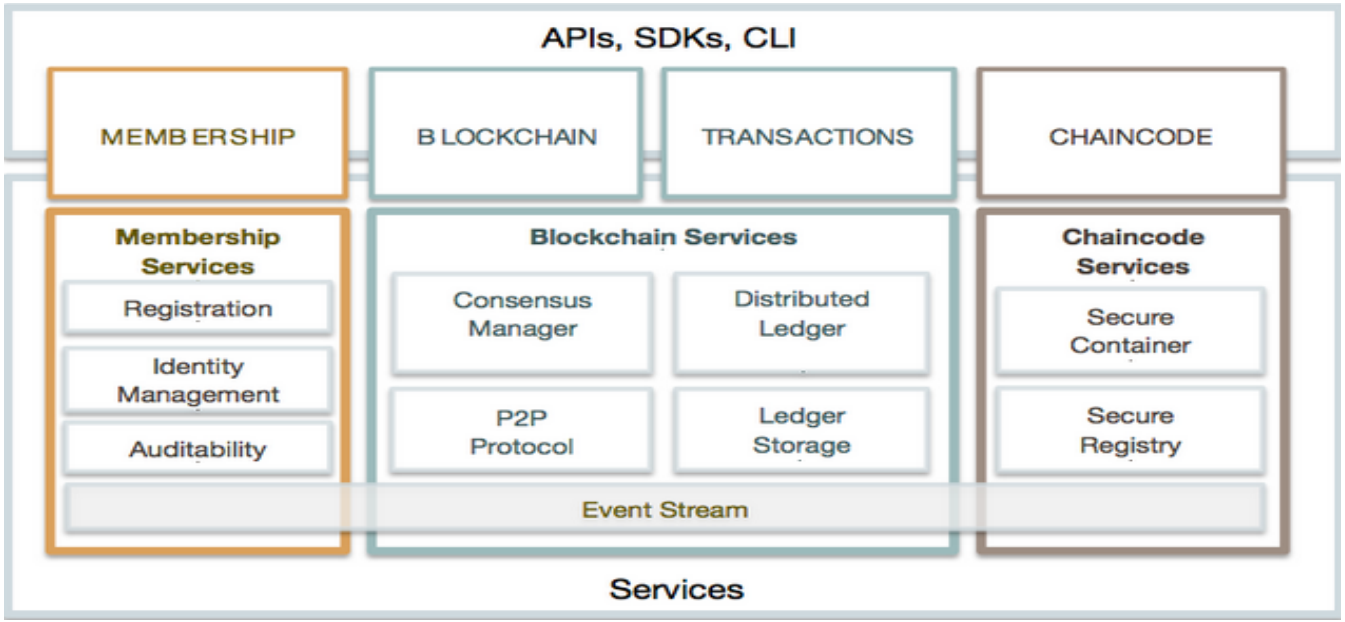



Assets
Participants
Transactions
Transaction
Functions
Access
Control
rules
Query
Definitions
Model File
.cto
Script File
.js
Access Control
.acl
Query File
.qry
Use Composer to create a Business Network Definition, comprised of Model (.cto), Script (.js), ACL (.acl) and Query (.qry) files.
Business Network Archive
.bna
Package up your Business Network Definition and export it as an archive (.bna file) ready to deploy it somewhere.
Hyperledger Fabric
Cloud / Local
Web Browser / Node.js
Online
Use ID Cards (which include connection profiles and credentials) to deploy your Business Network Definition to a distributed ledger.

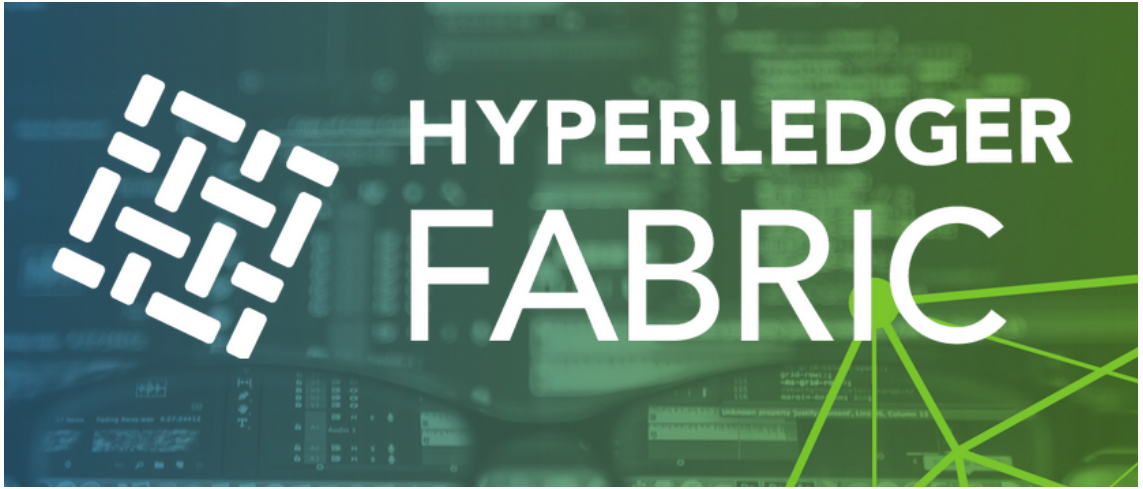


Deploying to Hyperledger Fabric and the Cloud

Hyperledger Fabric overview



- Hyperledger Fabric allows components, such as consensus and membership services, to be **plug-and-play**.
- Hyperledger Fabric leverages **container technology** (Docker) to host **smart contracts called "chaincode"** that comprise the application logic of the system.
- Chaincode is like Stored Procedures in the traditional database world

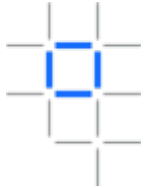


THE LINUX FOUNDATION

Hyperledger Fabric is an **open source blockchain framework** hosted by The Linux Foundation.

https://www.hyperledger.org/projects/fabric
https://github.com/hyperledger/fabric
https://hyperledger-fabric.readthedocs.io/en/release-1.1/

The role of the Cloud for Enterprise deployment



- Integration with existing enterprise systems
- Enterprise scale Performance
- High Availability and Global deployment
- Security and failover
- Backup and restore
- Connection to existing cloud-based enterprise business systems

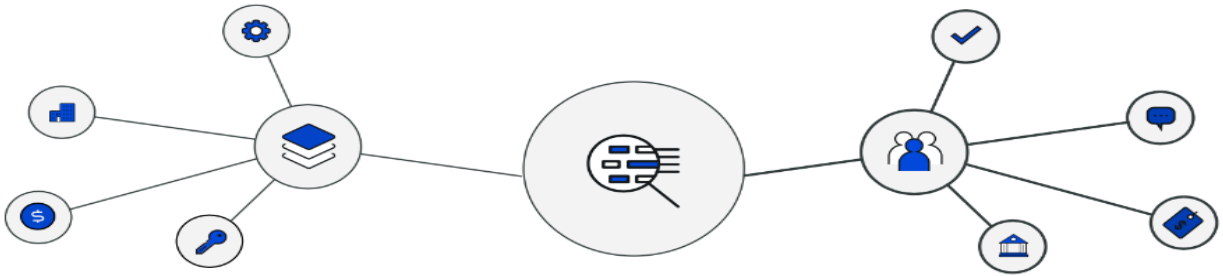
IBM Blockchain Starter Plan Beta

Blockchain /

Blockchain-BostonWed

Location: US South **Org:** Developer Advocacy **Space:** dev

Welcome back, alf



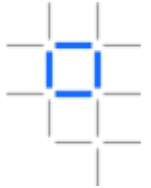
Use the Network Monitor to view and add network nodes, create channels, install chaincode applications, and test confidential transactions.

Enter Monitor

IBM

IBM

Overview



Overview

View and manage network resources for your organization. You can stop or start your resources and view the log file of the resource by selecting View Logs under "Actions". Learn more

Connection Profile

TYPE	NAME	STATUS
Orderer	orderer	Running
CA	org1-ca	Running
Peer	org1-peer1	Running

Members

View and manage the members (organizations) of the network on the Members tab. You can invite other organizations to the network or add more organizations in addition to the two organizations that you own by default. You can also view and manage admin certificates that are associated to your organizations on the Certificates tab. Learn more

Members | Certificates

Add Member

MEMBERS (2/16)	MSP ID	REQUESTER	STATUS	ACTION
Company A alf@us.ibm.com	org1	--	Joined	
Company B alf@us.ibm.com	org2	--	Joined	

Channels

You must have a channel to propose a transaction. If you're not a member of any channels, you can create one by clicking "Request Channel". Learn more

Search Channels

Request Channel

ID	TIME CREATED	BLOCK HEIGHT	PEERS	ACTIONS
defaultchannel	04/18/18 10:39 PDT	23	org1-peer1	⋮

* Channels created with the SDK cannot be edited here (yet)

APIs

Interact with the network by using APIs in the Swagger UI. You can also use the network credentials and integrate the APIs to your own application. Learn more

API reference list

Use the Swagger UI to try out the available catalog of REST APIs against the network.

Swagger UI

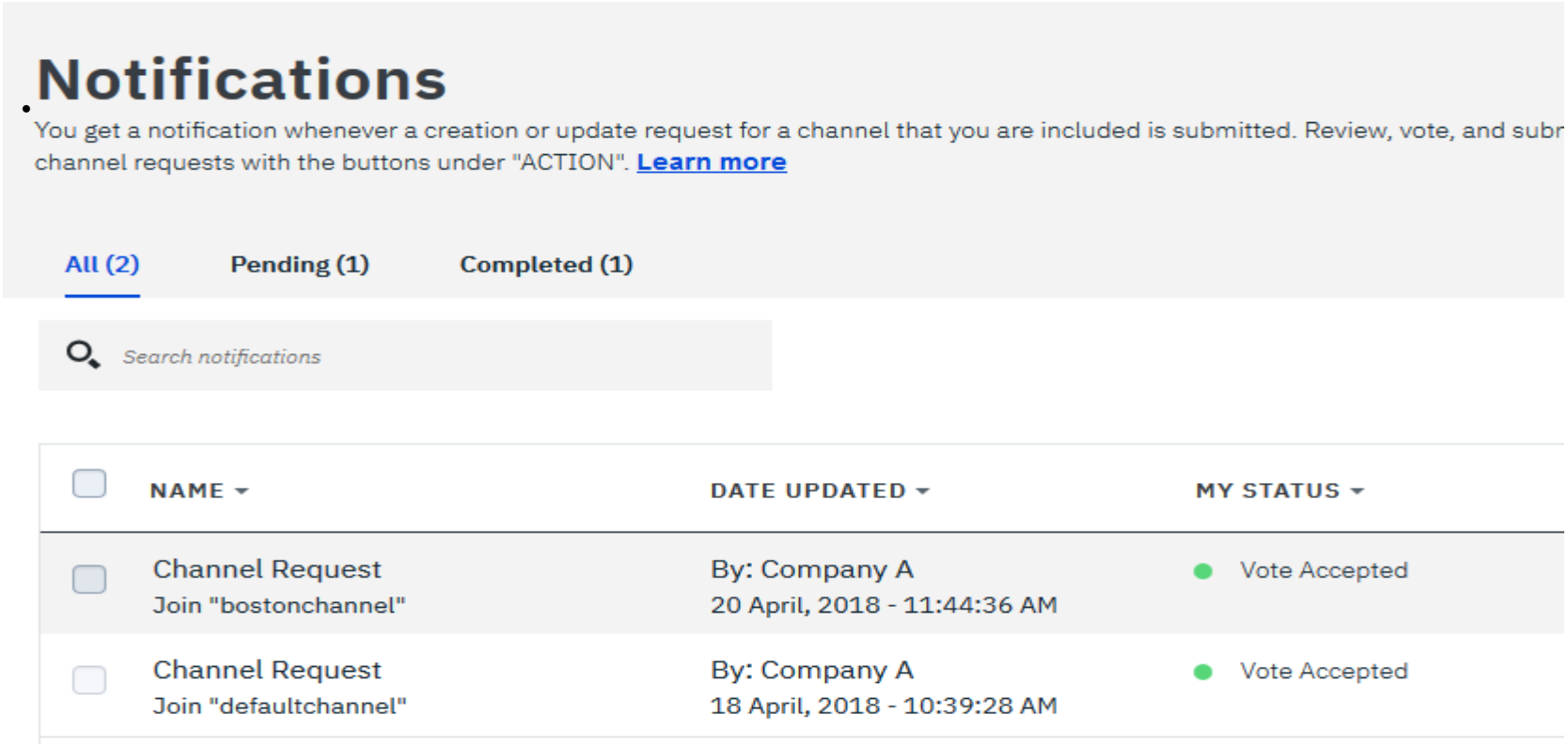
Network credentials

Use these network credentials to access the resource endpoints that your application needs to. Use "key" as your "Username" and "secret" as your "Password" when you authorize your APIs in the Swagger UI.


```
{
  "url": "https://ibmblockchain-starter.ng.bluemix.net",
  "network_id": "nabbe2cb650394e3794e233bd1b5153c3",
  "key": "org1",
  "secret":
}
```


Show secret

Notifications



Write code

You can develop your blockchain use case with IBM Blockchain Platform development environment, and then deploy the use case back to this network. **Learn more.**

Develop code

The IBM Blockchain Platform provides a free development environment based on industry standard tools and technologies (Hyperledger Composer, Javascript, Docker, Yeoman, and more) that you can use to model and code a ready-to-deploy use case.

You can trial this environment online or install it on your local machine. Learn more and download the tools at our website.

Visit website

Deploy code

After you create your blockchain use case, you can deploy it to this network in the cloud platform following the Deployment Guide.

Deployment guide

Install code

Chaincode must be installed on a peer. Select a peer and then install a chaincode on it. After you install the chaincode, you can request to instantiate it on a channel by clicking the Instantiate button under "Actions" for that chaincode. Clicking elsewhere on the chaincode will show you what channels the chaincode is instantiated on. Learn more. Do you already have a .bna file to deploy? See our deployment guide.

Choose peer...

Install Chaincode

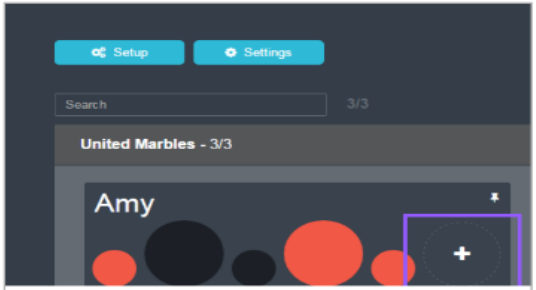
CODE ID	VERSION	ACTIONS
marbles_sample	3a30a0df1829	

Try samples

Sample applications provide you a better understanding of blockchain networks. You can also develop your own apps by installing our developer tools. This sample deployment approach leverages IBM Cloud Toolchains and helps to deploy sample applications, including chaincode, front-end apps, and a dev-ops pipeline in a single step. **Learn more**

Cloud Toolchain. Each sample deployment comes with source control, deployment pipeline, and the "chaincode" running on your blockchain network.

After you deploy the sample, any changes you push to the source control are automatically deployed to your Blockchain Platform service via the deployment pipeline.



Marbles

Deployed on 04/20/18

Toolchain Management

Remove

Launch

Vehicle Manufacture

- Track a new car from order to delivery
- Composer model and javaScript
- Angular web app

Deploy via Toolchain

View on GitHub

Build Proposal

The invocation needs to be signed and packaged as a proposal. This is done here, in the Marbles application.

Endorsement

Next, we send the transaction proposal to our peer for endorsement. The peer will simulate the transaction, verify the proposal, and then sign the proposal. It is then sent back to Marbles.

Ordering

Then Marbles will send our endorsed proposal to the orderer. The orderer will sequence transactions from throughout the network including ours.

Validate & Commit

The block containing our transaction is sent from the orderer to our peer. Finally it is validated by the peer and then committed to the peer's ledger.

* Note the speed of this process is slowed down for demonstration purposes

Invoke Complete

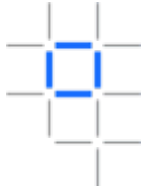
Close

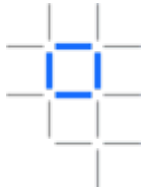
Support

Get help with your blockchain network. Learn more

Support | Release Notes

- GETTING STARTED
- COMMUNITY HELP
- SUPPORT TICKET
- BLOCKCHAIN SAMPLE APPLICATIONS
- HYPERLEDGER FABRIC
- HYPERLEDGER COMPOSER





Appendix