Tutorial A3: Deploying a smart contract

Estimated time: `10 minutes`

Now that we've created our first smart contract, we'd like to try it out. In this tutorial we will:

- Start an instance of a Hyperledger Fabric network in the local workspace
- Package the smart contract we previously created
- Deploy the smart contract to the running Hyperledger Fabric network

In order to successfully complete this tutorial, you must have first completed tutorial A2: Creating a smart contract. The project must be active in your VS Code workspace.

□   `A3.1`:   Expand the first section below to get started.

▶ **Start the VS Code Hyperledger Fabric environment**

□   `A3.2`:   Click on the IBM Blockchain Platform icon in the activity bar to show the blockchain side bar.
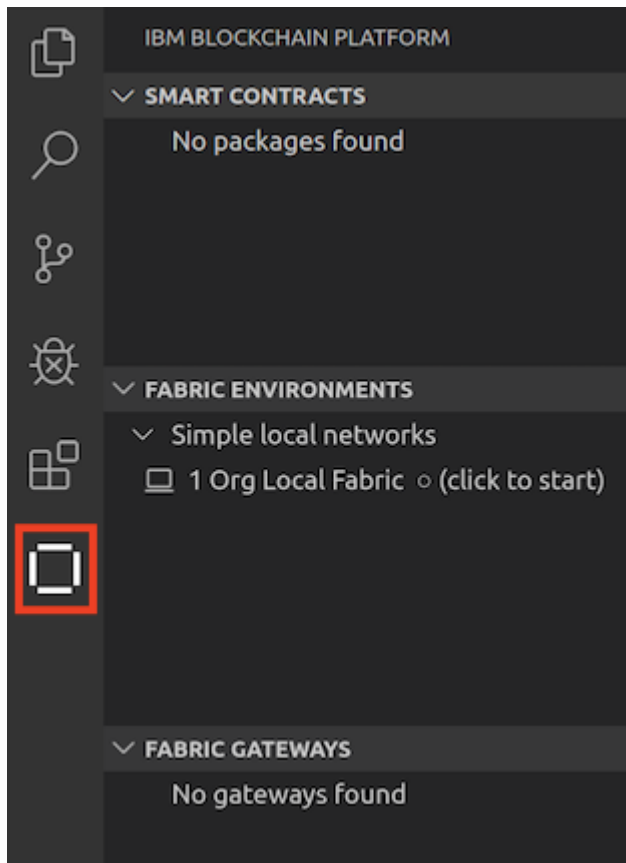
The Fabric Environments view

The IBM Blockchain Platform VS Code Extension helps you test your smart contracts in a Hyperledger Fabric network. The extension comes with a pre-configured one organization network that runs on your local machine ("1 Org Local Fabric"). You can connect to remote networks too; we will do this in a later tutorial.

The available networks are shown in the Fabric Environments view.
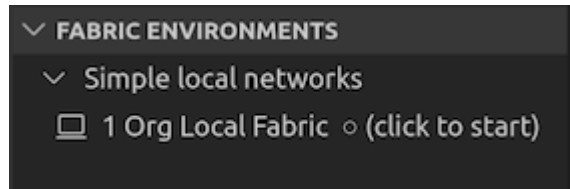
If you do not have a "1 Org Local Fabric" environment, then click + *Add local or remote environment* to create the environment. If creating the environment, click *Create new from template* and click *1 Org template*. Enter "1 Org local Fabric" as the name, and when you press *enter* the environment will start which may take up to 5 minutes.

We'll see later how this view also allows you to configure more realistic networks that also run entirely on your local machine. This allows you to check that your smart contract is functionally correct before you move to a more complex distributed network configuration.

The required Hyperledger Fabric components are automatically downloaded and started when you select it.

🖳 A3.3:   In the Fabric Environments view, click "*1 Org Local Fabric O (click to start)*"

This will download and start the embedded instance of Hyperledger Fabric, and may take up to five minutes to complete.
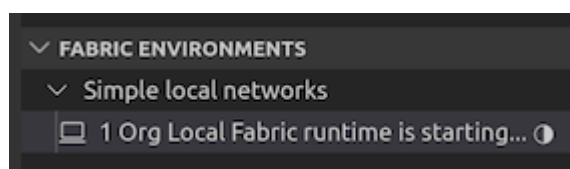
When Hyperledger Fabric has fully initialized, the view will change to show the channels, nodes and organizations in the local environment. You may need to click on the environment to see the details.
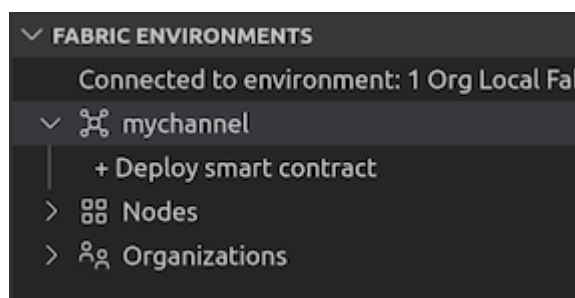
Each of these elements tells you what's configured in the connected environment:

- **Channels** define the scope of each network, and form one method of choosing how organizations share data. Deployed smart contracts that are available to the network will show under channels. We will look at channels in a later tutorial.

- **Nodes** are the Hyperledger Fabric components that make the system work. There are three types of nodes:

    - Peers which host ledgers and execute smart contracts
    - Orderers which assert transaction order and distribute blocks to peers
    - Certificate authorities which provide the means of identifying users and organizations on the network

- **Organizations** are the members of the blockchain network. Each organization will consist of many different users and types of users.

> For more about the components that make up a Hyperledger Fabric network, see the Hyperledger Fabric documentation.

If you expand the various sections you'll see the various defaults for each of these elements:

- Four **nodes**: a single peer called *Org1Peer1*, an ordering node called *Orderer* and a certificate authority for each of the two organizations.
- Two **organizations**, with identifiers of 'OrdererMSP' and 'Org1MSP'. The former will own the orderer and the latter the peer; it is good practice to use separate organizations for orderer nodes and peers.
- There is a single default network **channel** called *mychannel*.
- By default there are no **smart contracts** deployed to the channel.
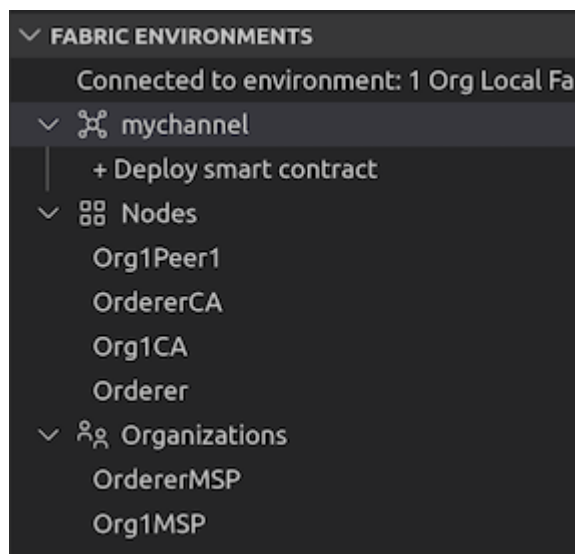
> **Starting again?**
> If you ever need to start with a new Hyperledger Fabric instance, hover over the Fabric Environments view, click the ellipsis ('...') and select 'Teardown Fabric Environment'. Use with caution: this will completely wipe the Hyperledger Fabric instance and anything deployed to it. Development files in your workspace (e.g. smart contract projects) will remain.

🔲 A3.4: Expand the next section of the tutorial to continue.

▶ Package and deploy the smart contract

We will now package and deploy our smart contract into the local environment. The VS Code extension has a simplified version of the process to deploy a smart contract and the default options provided work well for for simple projects such as this demo-contract.
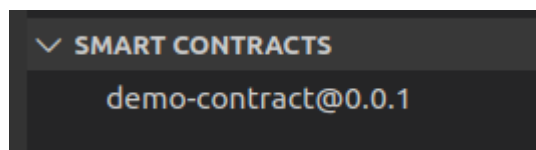
> **Want to know more?**
> For more about smart contract packages (chaincode) and the lifecycle, check out the Hyperledger Fabric documentation.
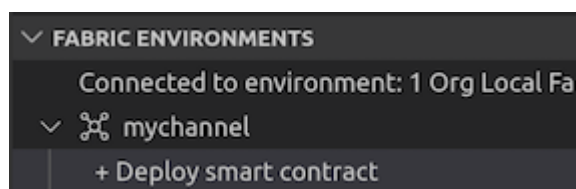
🔲   A3.5:   Move the mouse over the title bar of the Smart Contracts view, click the "..." that appears and select "Package Open Project".
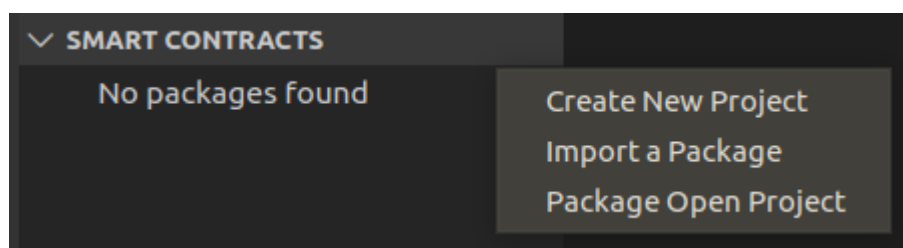
The Smart Contracts view will be updated to show the new package.

🔲   A3.6:   In the Fabric Environments view, click "mychannel" -> "+ Deploy smart contract".

🔲   A3.7:   In the Deploy Smart Contract form, select "demo-contract@0.0.1" from the drop down list, and click 'Next'.

A3.8: In step 2 of the form, default values for Definition name and version are provided, click 'Next' to move to Step 3 of the deploy.

For TypeScript smart contracts, both the name and version are taken from the *package.json* file in the root of the smart contract project.

The endorsement policy determines which peers get to run the smart contract. As we only have one peer in our organization, we can accept the default. Collections configuration and private data is an advanced technique for sharing data between organizations. We will not be using that feature for now; it is the subject of a later tutorial.

A3.9: In step 3 of the form, the automated steps of the deploy are sumarized, click 'Deploy' to start the deployment.
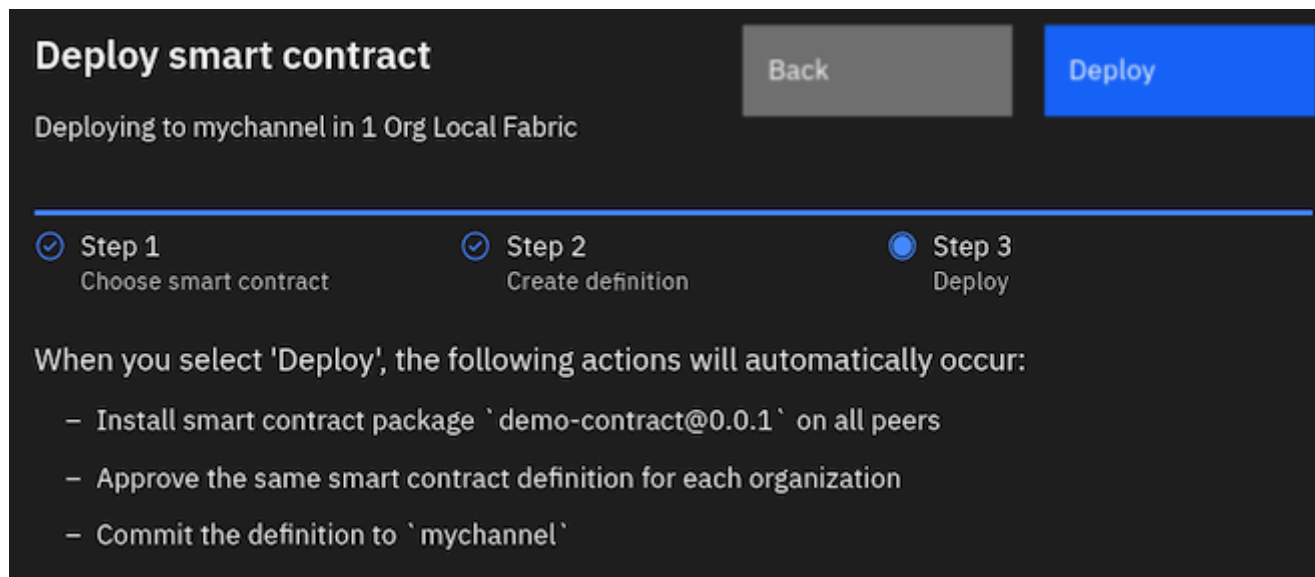
Deployment may take a few minutes to complete.

When deployment has completed you will see the new smart contract listed under "mychannel" in the Fabric Environments view.
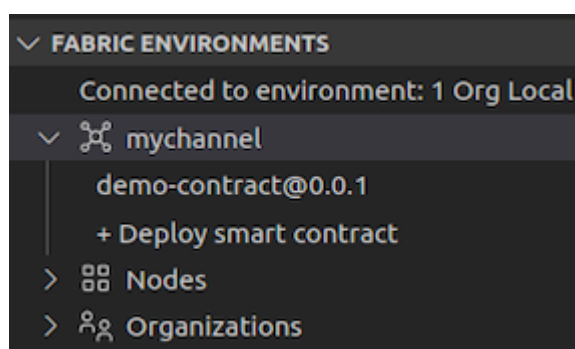
Congratulations! You've successfully deployed your first smart contract on a Hyperledger Fabric network. As we're going to see, there's much more to do - but this is a great start.

Summary

In this tutorial we started the built-in one organization Hyperledger Fabric network. We packaged our smart contract and deployed it.

In the next tutorial we will exercise the smart contract, to see how it behaves inside the network.

→ **Next: A4: Invoking a smart contract from VS Code**