Create a Cloud9 environment
IAM Configuration
Modify Cloud9 IAM role
▶ Set up a Fabric client
▼ Write and deploy chaincode
Chaincode development environment
Write chaincode
Create sharing policy
Create member identities
Configure main channel
Create main channel
Join main channel
Build chaincode
Install chaincode
Approve and commit the chaincode
▶ Test the chaincode
Congratulations
▼ **AWS account access**
Open AWS console (us-east-1)
Get AWS CLI credentials
Exit event

Event ends in 26 minutes.

Configure main channel

Only the **Retailer** should run the following command in its Cloud9 terminal to update the `configtx.yaml` channel configuration file.

Now that the chaincode has been thoroughly tested with local unit tests, it's time to install it on member peer nodes and test it in its native Fabric environment. Similar to the `Javascript` chaincode you deployed in Module 2, you will use the Fabric Chaincode Lifecycle to package, install, approve and commit this `Node.js` chaincode onto a channel shared between the Retailer and Supplier members.

You will recall that chaincode must be installed and committed to a channel by its members in order to be used, which means that we need to create a shared channel that both the Retailer and Supplier members belong to. This process create a channel configuration file with two organizations; this file will be named `configtx.yaml`.

```
cd
cat <<EOT > configtx.yaml
################################################################################
#
#   ORGANIZATIONS
#
#   This section defines the organizational identities that can be referenced
#   in the configuration profiles.
#
################################################################################
Organizations:
    # Retailer defines an MSP using the sampleconfig. It should never be used
    # in production but may be used as a template for other definitions.
    - &Retailer
        # Name is the key by which this org will be referenced in channel
        # configuration transactions.
        # Name can include alphanumeric characters as well as dots and dashes.
        Name: $RETAILERID
        # ID is the key by which this org's MSP definition will be referenced.
        # ID can include alphanumeric characters as well as dots and dashes.
        ID: $RETAILERID
        # SkipAsForeign can be set to true for org definitions which are to be
        # inherited from the orderer system channel during channel creation.  This
        # is especially useful when an admin of a single org without access to the
        # MSP directories of the other orgs wishes to create a channel.  Note
        # this property must always be set to false for orgs included in block
        # creation.
        SkipAsForeign: false
        Policies: &RetailerPolicies
            Readers:
                Type: Signature
```

Create a Cloud9 environment
IAM Configuration
Modify Cloud9 IAM role
▶ Set up a Fabric client
▼ Write and deploy chaincode
Chaincode development environment
Write chaincode
Create sharing policy
Create member identities
Configure main channel
Create main channel
Join main channel
Build chaincode
Install chaincode
Approve and commit the chaincode
▶ Test the chaincode
Congratulations
▼ **AWS account access**
Open AWS console (us-east-1)
Get AWS CLI credentials
Exit event

```
                Rule: "OR('Retailer.member', 'Supplier.member')"
                # If your MSP is configured with the new NodeOUs, you might
                # want to use a more specific rule like the following:
                # Rule: "OR('Retailer.admin', 'Retailer.peer', 'Retailer.client')"
            Writers:
                Type: Signature
                Rule: "OR('Retailer.member', 'Supplier.member')"
                # If your MSP is configured with the new NodeOUs, you might
                # want to use a more specific rule like the following:
                # Rule: "OR('Retailer.admin', 'Retailer.client')"
            Admins:
                Type: Signature
                Rule: "OR('Retailer.admin')"
        # MSPDir is the filesystem path which contains the MSP configuration.
        MSPDir: $HOME/retailer-admin-msp
        # AnchorPeers defines the location of peers which can be used for
        # cross-org gossip communication. Note, this value is only encoded in
        # the genesis block in the Application section context.
        AnchorPeers:
            - Host: 127.0.0.1
              Port: 7051
    - &Supplier
        Name: $SUPPLIERID
        ID: $SUPPLIERID
        SkipAsForeign: false
        Policies: &SupplierPolicies
            Readers:
                Type: Signature
                Rule: "OR('Supplier.member', 'Retailer.member')"
                # If your MSP is configured with the new NodeOUs, you might
                # want to use a more specific rule like the following:
                # Rule: "OR('Retailer.admin', 'Retailer.peer', 'Retailer.client')"
            Writers:
                Type: Signature
                Rule: "OR('Supplier.member', 'Retailer.member')"
                # If your MSP is configured with the new NodeOUs, you might
                # want to use a more specific rule like the following:
                # Rule: "OR('Retailer.admin', 'Retailer.client')"
            Admins:
                Type: Signature
                Rule: "OR('Supplier.admin')"
        # MSPDir is the filesystem path which contains the MSP configuration.
        MSPDir: $HOME/supplier-admin-msp
        # AnchorPeers defines the location of peers which can be used for
        # cross-org gossip communication. Note, this value is only encoded in
        # the genesis block in the Application section context.
        AnchorPeers:
            - Host: 127.0.0.1
              Port: 7052
################################################################################
#
#   CAPABILITIES
#
#   This section defines the capabilities of fabric network. This is a new
#   concept as of v1.1.0 and should not be utilized in mixed networks with
```

Create a Cloud9 environment
IAM Configuration
Modify Cloud9 IAM role
▶ Set up a Fabric client
▼ Write and deploy chaincode
Chaincode development environment
Write chaincode
Create sharing policy
Create member identities
Configure main channel
Create main channel
Join main channel
Build chaincode
Install chaincode
Approve and commit the chaincode
▶ Test the chaincode
Congratulations
▼ **AWS account access**
Open AWS console (us-east-1)
Get AWS CLI credentials
Exit event

```
#   v1.0.x peers and orderers.  Capabilities define features which must be
#   present in a fabric binary for that binary to safely participate in the
#   fabric network.  For instance, if a new MSP type is added, newer binaries
#   might recognize and validate the signatures from this type, while older
#   binaries without this support would be unable to validate those
#   transactions.  This could lead to different versions of the fabric binaries
#   having different world states.  Instead, defining a capability for a channel
#   informs those binaries without this capability that they must cease
#   processing transactions until they have been upgraded.  For v1.0.x if any
#   capabilities are defined (including a map with all capabilities turned off)
#   then the v1.0.x peer will deliberately crash.
#
################################################################################
Capabilities:
    # Channel capabilities apply to both the orderers and the peers and must be
    # supported by both.
    # Set the value of the capability to true to require it.
    # Note that setting a later Channel version capability to true will also
    # implicitly set prior Channel version capabilities to true. There is no need
    # to set each version capability to true (prior version capabilities remain
    # in this sample only to provide the list of valid values).
    Channel: &ChannelCapabilities
        # V2.0 for Channel is a catchall flag for behavior which has been
        # determined to be desired for all orderers and peers running at the v2.0.0
        # level, but which would be incompatible with orderers and peers from
        # prior releases.
        # Prior to enabling V2.0 channel capabilities, ensure that all
        # orderers and peers on a channel are at v2.0.0 or later.
        V2_0: true
    # Orderer capabilities apply only to the orderers, and may be safely
    # used with prior release peers.
    # Set the value of the capability to true to require it.
    Orderer: &OrdererCapabilities
        # V1.1 for Orderer is a catchall flag for behavior which has been
        # determined to be desired for all orderers running at the v1.1.x
        # level, but which would be incompatible with orderers from prior releases.
        # Prior to enabling V2.0 orderer capabilities, ensure that all
        # orderers on a channel are at v2.0.0 or later.
        V2_0: true
    # Application capabilities apply only to the peer network, and may be safely
    # used with prior release orderers.
    # Set the value of the capability to true to require it.
    # Note that setting a later Application version capability to true will also
    # implicitly set prior Application version capabilities to true. There is no need
    # to set each version capability to true (prior version capabilities remain
    # in this sample only to provide the list of valid values).
    Application: &ApplicationCapabilities
        # V2.0 for Application enables the new non-backwards compatible
        # features and fixes of fabric v2.0.
        # Prior to enabling V2.0 orderer capabilities, ensure that all
        # orderers on a channel are at v2.0.0 or later.
        V2_0: true
################################################################################
#
#   CHANNEL
```

Create a Cloud9 environment
IAM Configuration
Modify Cloud9 IAM role
Set up a Fabric client
Write and deploy chaincode
Chaincode development environment
Write chaincode
Create sharing policy
Create member identities
Configure main channel
Create main channel
Join main channel
Build chaincode
Install chaincode
Approve and commit the chaincode
Test the chaincode
Congratulations
AWS account access
Open AWS console (us-east-1)
Get AWS CLI credentials
Exit event

```
#
#   This section defines the values to encode into a config transaction or
#   genesis block for channel related parameters.
#
################################################################################
Channel: &ChannelDefaults
    # Policies defines the set of policies at this level of the config tree
    # For Channel policies, their canonical path is
    #   /Channel/<PolicyName>
    Policies:
        # Who may invoke the 'Deliver' API
        Readers:
            Type: ImplicitMeta
            Rule: "ANY Readers"
        # Who may invoke the 'Broadcast' API
        Writers:
            Type: ImplicitMeta
            Rule: "ANY Writers"
        # By default, who may modify elements at this config level
        Admins:
            Type: ImplicitMeta
            Rule: "MAJORITY Admins"
    # Capabilities describes the channel level capabilities, see the
    # dedicated Capabilities section elsewhere in this file for a full
    # description
    Capabilities:
        <<: *ChannelCapabilities
################################################################################
#
#   APPLICATION
#
#   This section defines the values to encode into a config transaction or
#   genesis block for application-related parameters.
#
################################################################################
Application: &ApplicationDefaults
    # Organizations is the list of orgs which are defined as participants on
    # the application side of the network
    Organizations:
    # Policies defines the set of policies at this level of the config tree
    # For Application policies, their canonical path is
    #   /Channel/Application/<PolicyName>
    Policies: &ApplicationDefaultPolicies
        LifecycleEndorsement:
            Type: ImplicitMeta
            Rule: "ANY Readers"
        Endorsement:
            Type: ImplicitMeta
            Rule: "ANY Readers"
        Readers:
            Type: ImplicitMeta
            Rule: "ANY Readers"
        Writers:
            Type: ImplicitMeta
            Rule: "ANY Writers"
```

Create a Cloud9 environment
IAM Configuration
Modify Cloud9 IAM role
▶ Set up a Fabric client
▼ Write and deploy chaincode
Chaincode development environment
Write chaincode
Create sharing policy
Create member identities
Configure main channel
Create main channel
Join main channel
Build chaincode
Install chaincode
Approve and commit the chaincode
▶ Test the chaincode
Congratulations
▼ **AWS account access**
Open AWS console (us-east-1)
Get AWS CLI credentials
Exit event

```
        Admins:
            Type: ImplicitMeta
            Rule: "MAJORITY Admins"

    Capabilities:
        <<: *ApplicationCapabilities
################################################################################
#
#   PROFILES
#
#   Different configuration profiles may be encoded here to be specified as
#   parameters to the configtxgen tool. The profiles which specify consortiums
#   are to be used for generating the orderer genesis block. With the correct
#   consortium members defined in the orderer genesis block, channel creation
#   requests may be generated with only the org member names and a consortium
#   name.
#
################################################################################
Profiles:
    TwoOrgChannel:
        <<: *ChannelDefaults
        Consortium: AWSSystemConsortium
        Application:
            <<: *ApplicationDefaults
            Organizations:
                - *Retailer
                - *Supplier
EOT
```

Previous Next