If running the nodejs sample via self-hosted Dapr mode, be sure you have Node.js installed on your machine, and run npm install to install the Node packages.

In this sample, I used an Azure Redis instance as the state store. You can use the Dapr built-in Redis provider locally, but if you want to use state store (or other providers) in Container Apps, you can’t use the built-in Redis provider. You will need to create the Azure Redis instance (or other provider of your choice) and configure it in the Components folder before moving forward.

Change to the directory that contains the application (eg the nodejs directory if Nodejs) and run the following:

dapr run --app-id myapp1 --dapr-http-port 3500 --app-port 5000 node app.js --components-path ./components

While Dapr is running, in a separate command prompt, make direct requests against the Dapr state endpoint to demonstrate writing to and reading from the statestore:

curl -X POST <http://localhost:3500/v1.0/state/mystatestore> -H "Content-Type: application/json" -d ["{\"key\": \"fakekey1\", \"value\": \"I like donuts\"}"]

curl <http://localhost:3500/v1.0/state/mystatestore/fakekey1> -H "Content-Type: application/json"

Now use the application to write to and read from the state store:

curl -X POST <http://localhost:5000/writestatestore> -H "Content-Type: application/json"

curl <http://localhost:5000/getstatestore> -H "Content-Type: application/json"

Note: The writestatestore and getstatestore endpoints are not Dapr endpoints. These are arbitrary names I gave these application routes. The code in these route methods is what makes the requests to the Dapr endpoints. The application writes the current long datetime as a value in the state store.

If you push the container image to Azure Container Registry, will need to set the ACR to use anonymous access for now in order to work with Container Apps:

az acr update --name *myregistry* --anonymous-pull-enabled

To build the container:

docker build -t *myregistry*.azurecr.io/statestorenodejs:latest .

To authenticate to ACR:

Az login

az acr login -n *myregistry*

To push the container images to the registry:

docker push *myregistry*..azurecr.io/statestorenodejs:latest

placeholder for the moment since there are caveats with creating Container App environments at the moment.

To deploy to a Container App, first change the values in the parameters.json, under the deploy folder. If you use a provider other than Azure Redis, you will need to modify the Components section of the deploy.json file. Then in the command line, change to the deploy folder and run the following command:

az deployment group create --name ContainerAppStateStoreDeployment --resource-group *resourcegroup* --template-file deploy.json --parameters parameters.json

You can then test the writestatestore and getstatestore endpoints on the Container App. Replace localhost with the Container App FQDN.