

This document describe hows to use **Newman** run EdegeX blackbox tests.

## Prerequisite

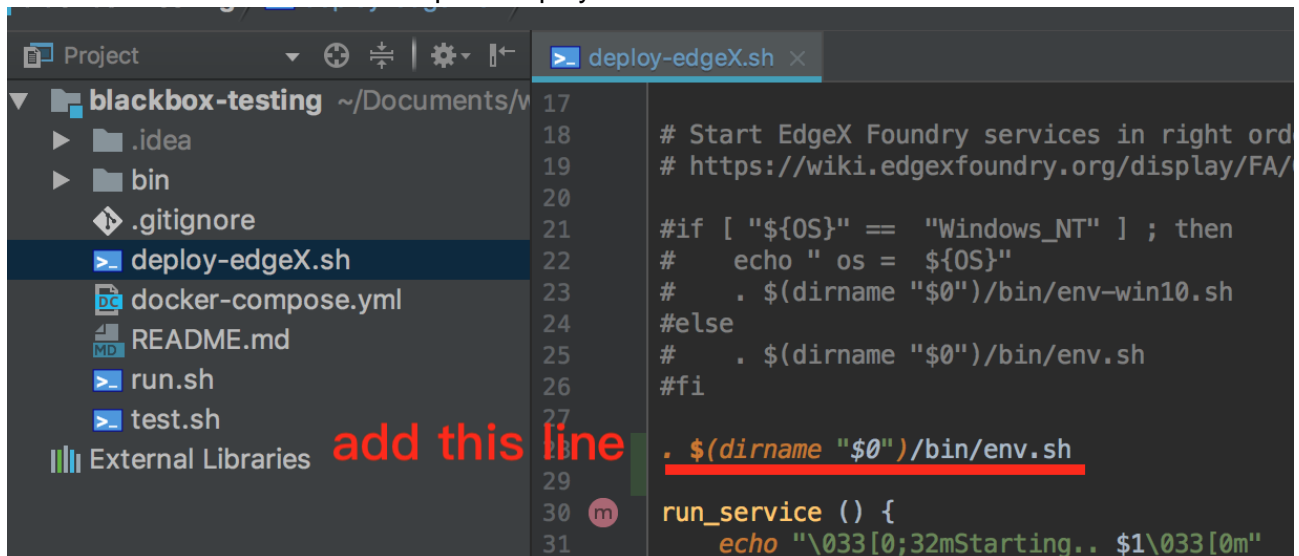
1. Clone blackbox test script, <https://github.com/edgexfoundry/blackbox-testing>
2. Install Newman app , `npm install newman --global` ,  
<https://github.com/postmanlabs/newman>

## Test a service's api with the blackbox-testing script

The blackbox-testing script uses Newman to run tests, but it combines all of the detailed steps.

### 1.deploy

change directory to `path/to/blackbox-testing`, add code in `./deploy-edgex.sh`, this code snippet can feed env variable for docker-compose deploy

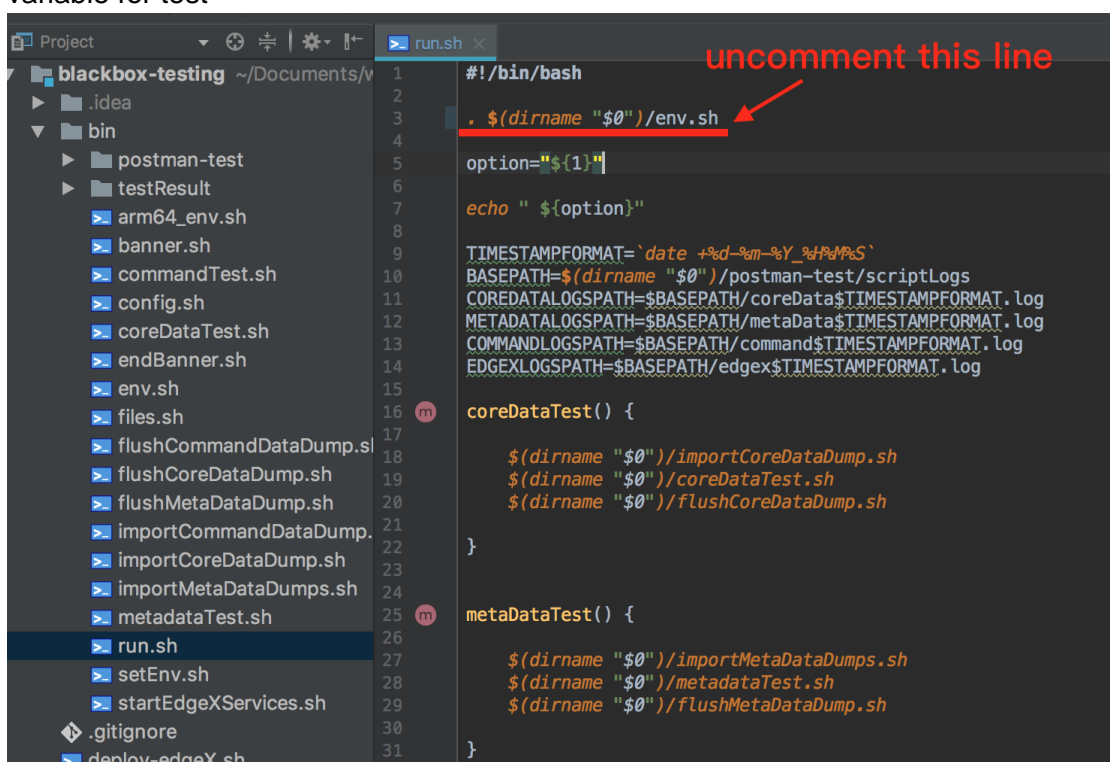


```
17
18 # Start EdgeX Foundry services in right ord
19 # https://wiki.edgexfoundry.org/display/FA/
20
21 #if [ "${OS}" == "Windows_NT" ] ; then
22 #   echo " os = ${OS}"
23 #   . $(dirname "$0")/bin/env-win10.sh
24 #else
25 #   . $(dirname "$0")/bin/env.sh
26 #fi
27
28 . $(dirname "$0")/bin/env.sh
29
30 run_service () {
31     echo "\033[0;32mStarting.. $1\033[0m"
```

then we can deploy by command line `sh deploy-edgex.sh`

### 2. run test

after startup edgex service , uncomment code in `bin/run.sh`, this code snippet can feed env variable for test



```
1 #!/bin/bash
2
3 . $(dirname "$0")/env.sh
4
5 option="${1}"
6
7 echo " ${option}"
8
9
10 TIMESTAMPFORMAT=`date +%d-%m-%Y_%H%M%S`
11 BASEPATH=$(dirname "$0")/postman-test/scriptLogs
12 COREDATALOGSPATH=$BASEPATH/coreData$TIMESTAMPFORMAT.log
13 METADATALOGSPATH=$BASEPATH/metaData$TIMESTAMPFORMAT.log
14 COMMANDLOGSPATH=$BASEPATH/command$TIMESTAMPFORMAT.log
15 EDGEXLOGSPATH=$BASEPATH/edgex$TIMESTAMPFORMAT.log
16
17 coreDataTest() {
18     $(dirname "$0")/importCoreDataDump.sh
19     $(dirname "$0")/coreDataTest.sh
20     $(dirname "$0")/flushCoreDataDump.sh
21 }
22
23
24 metaDataTest() {
25     $(dirname "$0")/importMetaDataDumps.sh
26     $(dirname "$0")/metadataTest.sh
27     $(dirname "$0")/flushMetaDataDump.sh
28 }
29
30
31 }
```

than we can:

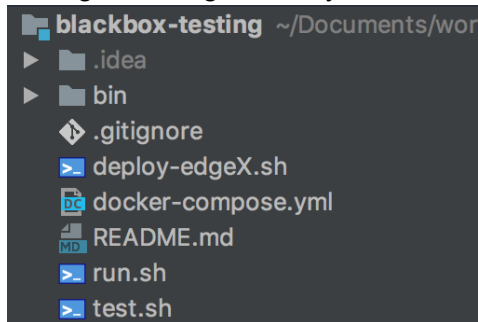
1. run all test by `sh ./bin/run.sh -all`
2. run CoreData test by `sh ./bin/run.sh -cd`
3. run metaData test by `sh ./bin/run.sh -md`
4. run command test by `sh ./bin/run.sh -co`

## Test a service's api with Newman

Below are the detailed instructions that describe how the blackbox-testing script works.

For example, when we want to test coredata's event api by running the tests with Newman.

1. Copy test script to edgex-mongo for import test data
  - a. Change working directory to "blackbox-testing" you had download



- b. execute command , `docker cp ./bin/postman-test/. edgex-mongo:/postman-test`
    - c. check execute result , `docker exec -t edgex-mongo /bin/bash -c "ls /postman-test"`

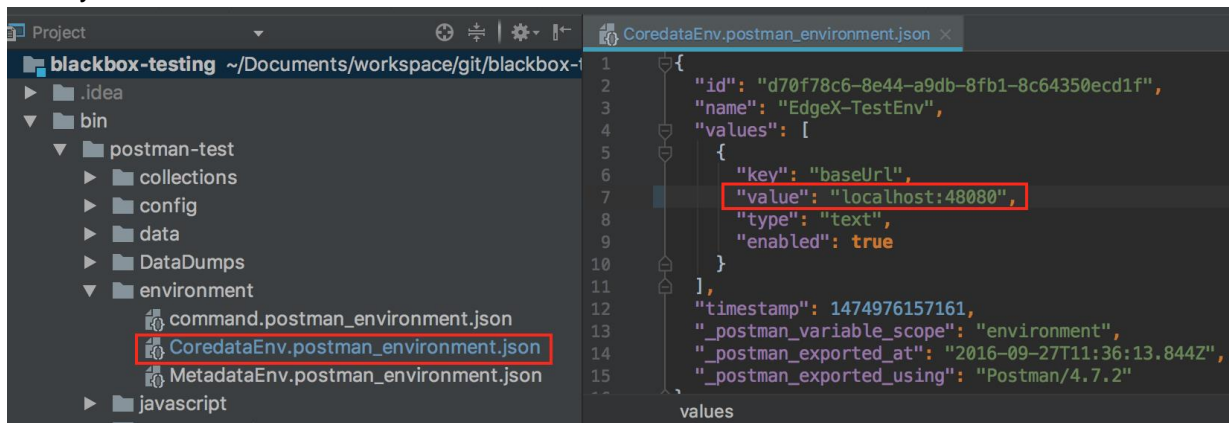
```
Bruce-MacBook-Pro:~ bruce$ docker exec -t edgex-mongo /bin/bash -c "ls /postman-test"
DataDumps    data          sample_profile.yaml
collections  environment   sample_profile_duplicated_command.yaml
config       javascript    sample_profile_empty.yaml
```

2. Import coredata's test data, run below commands

- a. `docker exec -t edgex-mongo /bin/bash -c "mongoimport -d coredata -c event --file /postman-test/DataDumps/coredata/eventDb.json && mongoimport -d coredata -c reading --file /postman-test/DataDumps/coredata/readingDb.json && mongoimport -d coredata -c valueDescriptor --file /postman-test/DataDumps/coredata/valueDescriptorDb.json"`

3. Test coredata's reading api

- b. modify environment variable



- c. Execute below command
      - i. `newman run ./bin/postman-test/collections/coredata.postman_collection.json --folder event -e ./bin/postman-test/environment/CoredataEnv.postman_environment.json -d ./bin/postman-test/data/eventData.json`

d. And will see the results as shown the following picture

	executed	failed
iterations	1	0
requests	15	0
test-scripts	15	0
prerequisite-scripts	0	0
assertions	57	7
total run duration: 1309ms		
total data received: 12.25KB (approx)		
average response time: 45ms		
# failure	detail	
1. AssertionError	expected false to be truthy at assertion:0 in test-script inside "event / 95 http://localhost:48080/api/v1/event "	
2. AssertionError	expected false to be truthy at assertion:0 in test-script inside "event / 15 http://localhost:48080/api/v1/event/id/:id"	
3. AssertionError	expected false to be truthy at assertion:0 in test-script inside "event / 15 http://localhost:48080/api/v1/event/id/:id"	

4. After coredata's test done, flush coredata's test data, run the command below

- e. `docker exec -t edgex-mongo /bin/bash -c "mongo coredata /postman-test/javascript/coredata/event.js && mongo coredata /postman-test/javascript/coredata/reading.js && mongo coredata /postman-test/javascript/coredata/valueDescriptor.js"`

## Notice:

1.If you follow <https://wiki.edgexfoundry.org/display/FA/Get+EdgeX+Foundry++Users> to install Edgex, when the tests are complete you can run command below to uninstall Edgex:

1. docker-compose down (this command will remove all container)
2. docker volume prune (this command will clean old test data)

2.When using the Newman command tool you can pass different parameters to test different api collections, just follow below command format and replace the parameters .

format:

```
newman run <collection-file-path> --folder <particular-folder-in-a-collection> -e <environment-file-path> -d <test-data-file-path>
```

## Coredata API

**<collection-file-path>** : collections/core-metadata.postman\_collection.json

**<environment-file-path>** : environment/MetadataEnv.postman\_environment.json

api collections	<particular-folder-in-a-collection>	<test-data-file-path>
event	event	data/eventData.json
event error 4xx	event_error_4xx	data/eventData.json
reading	reading	data/readingData.json
reading error 4xx	reading_error_4xx	data/readingData.json
value descriptor	valuedescriptor	data/valueDescriptorData.json
value descriptor error 4xx	valuedescriptor_error_4xx	data/valueDescriptorData.json

## Metadata API

**<collection-file-path>** : collections/core-metadata.postman\_collection.json

**<environment-file-path>** : environment/MetadataEnv.postman\_environment.json

api collections	<particular-folder-in-a-collection>	<test-data-file>
addressable	addressable	data/addressableData.json
addressable error 4xx	addressable_error_4xx	data/addressableData.json
command	command	data/commandData.json
command error 4xx	command_error_4xx	data/commandData.json
device	device	data/deviceData.json
device error 4xx	device_error_4xx	data/deviceData.json
device profile	deviceprofile	data/deviceProfileData.json
device profile error 4xx	deviceprofile_error_4xx	data/deviceProfileData.json
device report	devicereport	data/deviceReportData.json
device report error 4xx	devicereport_error_4xx	data/deviceReportData.json
device service	deviceservice	data/deviceServiceData.json
device service error 4xx	deviceservice_error_4xx	data/deviceServiceData.json
provision watcher	provisionwatcher	data/provisionWatcherData.json
provision watcher error 4xx	provisionwatcher_error_4xx	data/provisionWatcherData.json
schedule	schedule	data/scheduleData.json

schedule error 4xx	schedule_error_4xx	data/scheduleData.json
schedule event	scheduleevent	data/scheduleEventData.json
schedule event error 4xx	scheduleevent_error_4xx	data/scheduleEventData.json

## Cammand API

**<collection-file-path>** : collections/core-command.postman\_collection.json

**<environment-file-path>** : environment/command.postman\_environment.json

api collections	<particular-folder-in-a-collection>	<test-data-file-path>
device	device	data/coreCommandData.json
device error 4xx	device_error_4xx	data/coreCommandData.json