var amqp = require('amqplib');

amqp.connect('amqp://guest:guest@172.16.15.52:5672').then(function(conn) {

process.once('SIGINT', function() { conn.close(); });

return conn.createChannel().then(function(ch) {

var ok = ch.assertQueue('iot/logs', {durable: false});

ok = ok.then(function(\_qok) {

return ch.consume('iot/logs', function(msg) {

console.log(" [x] Received '%s'", msg.content.toString());

}, {noAck: true});

});

return ok.then(function(\_consumeOk) {

console.log(' [\*] Waiting for messages. To exit press CTRL+C');

});

});

}).catch(console.warn);

/\*

Copyright 2018 University of Salerno.

Licensed under the Apache License, Version 2.0 (the "License");

you may not use this file except in compliance with the License.

You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software

distributed under the License is distributed on an "AS IS" BASIS,

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and

limitations under the License.

\*/

var args = process.argv.slice(2);

console.log(args);

var amqp = require('amqplib');

for (var i = 0; i < args[0]; i++) {

var key = 'temperature';

amqp.connect('amqp://guest:guest@172.16.15.52:5672').then(function(conn) {

return conn.createChannel().then(function(ch) {

var ex = 'iot/sensors';

var ok = ch.assertExchange(ex, 'topic', {durable: false});

return ok.then(function() {

var message = Math.floor(Math.random()\*20);

ch.publish(ex, key, Buffer.from(message.toString()));

console.log(" [x] Sent %s:'%s'", key, message.toString());

return ch.close();

});

}).finally(function() { conn.close(); })

}).catch(console.log);

}

/\*

Copyright 2018 University of Salerno.

Licensed under the Apache License, Version 2.0 (the "License");

you may not use this file except in compliance with the License.

You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software

distributed under the License is distributed on an "AS IS" BASIS,

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and

limitations under the License.

\*/

var amqp = require('amqplib');

amqp.connect('amqp://guest:guest@172.16.15.52:5672').then(function(conn) {

process.once('SIGINT', function() { conn.close(); });

return conn.createChannel().then(function(ch) {

var ok = ch.assertQueue('iot/logs', {durable: false});

ok = ok.then(function(\_qok) {

return ch.consume('iot/logs', function(msg) {

console.log(" [x] Received '%s'", msg.content.toString());

}, {noAck: true});

});

return ok.then(function(\_consumeOk) {

console.log(' [\*] Waiting for messages. To exit press CTRL+C');

});

});

}).catch(console.warn);

apiVersion: "nuclio.io/v1"

kind: Function

metadata:

name: amqpconsume

namespace: nuclio

spec:

handler: "main:handler"

description: "Function the is called when a new message is arrived on the iot/sensors/temperature queue, //the function send back a feedback on the iot/logs queue."

runtime: nodejs

image: "nuclio/processor-amqpconsume:latest"

minReplicas: 1

maxReplicas: 1

targetCPU: 75

triggers:

amqp:

class: ""

kind: rabbit-mq

url: "amqp://guest:guest@172.16.15.52:5672"

attributes:

exchangeName: iot/sensors

queueName: iot/sensors

topics:

- temeprature

build:

functionSourceCode: JvbUNoYXJDb2RlKGFycmF5W2ldKSk7CiAgICAgICAgICB9CiAgICAgICAgICByZXR1cm4gcmVzdWx0OwogICAgICAgIH0KCiAgICAgICAgZXhwb3J0cy5oYW5kbGVyID0gZnVuY3Rpb24oY29udG2soIkludm9rZWQgRnVuY3Rpb24gIitGVU5DVElPTl9OQU1FKyIgb24gIitfZGF0YSk7CiAgICAgICAgfTs=

commands:

- 'npm install amqplib'

codeEntryType: sourceCode

platform: {}

apiVersion: "nuclio.io/v1"

kind: Function

metadata:

name: amqpevent

namespace: nuclio

spec:

handler: "main:handler"

description: "Function that generate an event on the AMQP queue sending a temperature value."

runtime: nodejs

image: "nuclio/processor-amqpevent:latest"

minReplicas: 1

maxReplicas: 1

targetCPU: 75

build:

functionSourceCode: dmFyIGFtcXAgPSByZXF1aXJlKCdhbXFwbGliJyk7CgpleHBvcnRzLmhhbmRsZXIgPSBmdW5jdGlvbihjb250ZXh0LCBldmVudCkgewogICAgdmFyIGtleSA9ICd0ZW1wZXJhdHVyZSc7CiAgICB2YXIgbWVbmFsbHkoZnVuY3Rpb24oKSB7IGNvbm4uY2xvc2UoKTsgIH0pCiAgICB9KS5jYXRjaChjb25zb2xlLmxvZyk7CiAgICAKICAgIGNvbnRleHQuY2FsbGJhY2soJ3NlbmQgJyttZXNzYWdlKTsKfTs=

commands:

- 'npm install amqplib'

codeEntryType: sourceCode

platform: {}

apiVersion: "nuclio.io/v1"

kind: Function

metadata:

name: mqttconsume

namespace: nuclio

spec:

handler: "main:handler"

description: "Function the is called when a new message is arrived on the iot/sensors/temperature queue, //the function send back a feedback on the iot/logs queue."

runtime: nodejs

image: "nuclio/processor-mqttconsume:latest"

minReplicas: 1

maxReplicas: 1

targetCPU: 75

triggers:

myMqttTrigger:

kind: "mqtt"

url: "guest:guest@172.16.15.52:1883"

attributes:

subscriptions:

- topic: iot/sensors/temperature

qos: 0

build:

functionSourceCode: dmFyIGFtcXAgPSByZXF1aXJlKCdhbXFwbGliJyk7CiAgICAgICAgdmFyIEZVTkNUSU9OX05BTUUgPSAibXF0dGNvbnN1bWUiOwoZXZlbnQuYm9keS5kYXRhKTsKCiAgICAgICAgICAgIGNvbnRleHQuY2FsbGJhY2soImZlZWRiYWNrICIrX2RhdGEpOwoKICAgICAgICAgICAgY29uc29sZS5sb2coIlRSSUdHRVIgIitfZGF0YSk7CiAgICAgICAgICAgIHNlbmRfZmVlZGJhY2soIkludm9rZWQgRnVuY3Rpb24gTVFUVDogIitGVU5DVElPTl9OQU1FKyIgcmVjZWl2ZWQgIitfZGF0YSk7CiAgICAgICAgfTs=

commands:

- 'npm install amqplib'

codeEntryType: sourceCode

platform: {}

apiVersion: "nuclio.io/v1"

kind: Function

metadata:

name: mqttrandomtemperature

namespace: nuclio

spec:

handler: "main:handler"

description: "Function to generate an event on the MQTT queue sending a temperature value."

runtime: nodejs

image: "nuclio/processor-mqtt-random-temperature:latest"

minReplicas: 1

maxReplicas: 1

targetCPU: 75

build:

functionSourceCode: dmFyIG1xdHQgPSByZXF1aXJlKCdtcXR0JyksIHVybCA9IHJlcXVpcmUoJ3VybCcpOwoKdmFyIG1xdHRfdXJsID0gdXJsLnBhcnNlKHByb2Nlc3MuZW52LkNMT1VEQU1RUF9NUVRUX1VSTCB8fCAnbXF0dDovL2d1ZXN0Omd1ZXN0QDE3Mi4xNi4x

commands:

- 'npm install mqtt'

codeEntryType: sourceCode

platform: {}