Change this code as you deem fit, for your TBB. You can add variables, methods, sensor integrations, etc here.

Change method bodies in this code, **but do not change the method names and signatures** (these are being called from elsewhere).

**Do not change** any code here, this is standardized code across all TBBs that integrates with the IoT Library and cloud.

#include<StubCopperCube.h>

#include <EEPROM.h>

/\* #### Update code version for every committed change #### \*/

const char\* codeName = "sample tbb stub";

const char\* codeVersion = "0.1";

/\* ################################################ \*/

/\* ##### block1 starts : change as appropriate ##### \*/

// global variables needed by this tbb are declared here

// eg: assign GPIO pins

int relayPin = 14;

/\* ##### block1 ends ######################### \*/

/\* ##### block2 starts : do not change this code ##### \*/

// standard code needed to make this code re-usable across all tbb(s)

boolean debug = true; // false for PROD

CopperCube cube(codeName, codeVersion, debug);

// common set-up required for device to work as IoT Cube

void setupCube() {

cube.initDevice(true);

cube.setMqttCallback(receiveMessage); // receiveMessage() is called when there is an incoming mqtt packet

cube.connect();

}

// common cube and device-specific set-up code goes here

void setup() {

setupCube();

executeDeviceSetup(); // device-specific setup outsourced to this method

}

void loop() {

cube.loop(); delay(100);

executeDeviceLoop(); // device-specific loop code outsourced to this method

}

/\* ##### block2 ends : do not change this code ##### \*/

**Default MQTT Topics for send<> methods:**

sendData() -> <deploymentid>/cube/data

sendEvent() -> <deploymentid>/cube/event

sendState() -> <deploymentid>/cube/state

sendException() -> <deploymentid>/cube/exception

/\* ##### block3 starts : change method code as needed by tbb ##### \*/

// do not change method names and signatures in this block, but method body can be changed

// device-specific set-up code goes here.

void executeDeviceSetup() {

// some setup here, example: pin mode etc

// eg:

pinMode(relayPin, OUTPUT);

// optional, if any extra topics (in addition to command) are required

cube.subscribeToTopic("anyothertopic");

delay(1000);

}

// device-specific loop code goes here. modify this in any way required including method signature

void executeDeviceLoop() {

// some device-specific loop code here

cube.sendData("temp", 60.88);

//cube.sendEvent("door", "open");

delay(1000);

}

// callback method for incoming mqtt packets. this method is only called by library of the packett is addressed to this device

void receiveMessage(char\* topic, const JsonDocument& payload) {

// print incoming message:

serializeJson(payload, Serial);

//read payload items from JsonDocument object:

const char\* cmd = payload["command"];

const char\* mt = payload["msgtype"];

// check if a particular key is present in the payload:

if(cmd == NULL) {Serial.println("No command attribute");}

else if(strcmp(cmd, "on")){

Serial.println(cmd);

// switch relay on

}

}

/\* ##### block3 ends : change method code as needed by tbb ##### \*/

/\* ## block4 starts : add/delete/modify methods as needed by this tbb ## \*/

void sendDifferentKindsOfMessages() {

// example messages

// 3rd argument in each of the following is optional. each message has a default topic, if left blank

cube.sendEvent("doorevent", "event text", "eventTopic");

cube.sendData("depth", readIRSensor(), "dataTopic"); cube.sendState("switchstate", "on", "stateTopic"); cube.sendException("excepTopic", "exceptionType", "exception");

}

void sendCustomMessage() {

StaticJsonDocument<256> doc;

doc["msgtype"] = "new";

doc["newtype"] = "number";

doc["newval"] = 1351824120;

const char\* newTopic = "cube/message";

cube.sendGenericMessage(doc, newTopic); // no default topic; must be provided

}

/\* ## block4 ends : add/delete/modify methods as needed by this tbb ## \*/