endpoints.h

July 13th 2015

Contents

1 Description **1**

2 Add a point **1**

2.1 Basics 1

2.2 Interact with the Arduino 2

2.3 Return types 2

2.4 Example 2

1 Description

In our case, we run a COAP server on the Arduino. This server can send a response to any COAP request which the arduino receives through the LoRa Network. So, we can imagine a light sensor which send its value when we make a GET request. In the following documentation, the name of the Arduino object will be *toto.s.ackl.io*.

2 Add a point

2.1 Basics

To add a point, we need 2 things. First, we need a path to access this point. We declare the path as following:

```
//First create a coap_endpoint_path_t
static const coap_endpoint_path_t our_path = {1, {"example"}};
//1 is the length of the dictionnary ({"example"})

//Then, add this path to the endpoints
const coap_endpoint_t endpoints[] =
{
    {COAP_METHOD_GET, handle_get_well_known_core, &path_well_known_core, "ct=40"},
    {COAP_METHOD_GET, handle_example, &our_path, "ct=0"},
    {(coap_method_t)0, NULL, NULL, NULL}
};
```

With this code, we will match the path *toto.s.ackl.io/example* and start handle_example. We must to create the handle:

```
static int handle_example(coap_rw_buffer_t *scratch, const coap_packet_t *inpkt,
coap_packet_t *outpkt, uint8_t id_hi, uint8_t id_lo)
{
  Serial.print("Example!");
  char lightMsg[] = "Example!";

  return coap_make_response(scratch, outpkt, (const uint8_t *)lightMsg,
  strlen(lightMsg), id_hi, id_lo, &inpkt->tok, COAP_RSPCODE_CONTENT,
  COAP_CONTENTTYPE_TEXT_PLAIN);
}
```

2.2 Interact with the Arduino

The file endpoints.h include the Arduino library. So, you can write Arduino code in the handle without any problem. For example, we can imagine a handle to turn on a led:

```
static const coap_endpoint_path_t path_led_on = {1, {"ledon"}};

static int handle_get_led_on(coap_rw_buffer_t *scratch, const coap_packet_t *inpkt,
coap_packet_t *outpkt, uint8_t id_hi, uint8_t id_lo)
{
  Serial.println("led on!");
  digitalWrite(led, HIGH);
  char ledonmsg[] = "the led is on";

  return coap_make_response(scratch, outpkt, (const uint8_t *)ledonmsg,
  strlen(ledonmsg), id_hi, id_lo, &inpkt->tok, COAP_RSPCODE_CONTENT,
  COAP_CONTENTTYPE_TEXT_PLAIN);
}
```

2.3 Return types

With the previous examples, we send a text response. But we can respond other things:

- COAP_CONTENTTYPE_NONE
- COAP_CONTENTTYPE_TEXT_PLAIN
- COAP_CONTENTTYPE_APPLICATION_LINKFORMAT
- COAP_CONTENTTYPE_APPLICATION_XML

2.4 Example

To send a webpage:

```
static const coap_endpoint_path_t path_helloWorld = {1, {"helloWorld"}};

static int handle_get_helloWorld(coap_rw_buffer_t *scratch, const coap_packet_t *inpkt,
coap_packet_t *outpkt, uint8_t id_hi, uint8_t id_lo)
{
  Serial.println("hello World!");

  char hellomsg[] = "<html xmlns=\"http://www.w3.org/1999/xhtml\"><center><h1>
  Hello World :) </h1></center></html>";

  return coap_make_response(scratch, outpkt, (const uint8_t *)hellomsg, strlen(hellomsg),
  id_hi, id_lo, &inpkt->tok, COAP_RSPCODE_CONTENT, COAP_CONTENTTYPE_APPLICATION_XML);
}
```