**Protocol Design Draft:** Communication with IoT Device (Light bulb)

Using UDP sockets, you will write a simplified communication program between a server and a client.. The server will be responsible for listening to requests from the client. The client will send a request to the server which will consist of one of four possible actions (Listed below)..The client and server communicate using the message format specified in this document.

The client will perform the following functions:

1. Read in 3-4 arguments from the command line:
   1. IP address of server (127.0.0.1)
   2. Port of server (e.g. 12001)
   3. Action type number (e.g. 3) (Explained later)
   4. Color (e.g. Red) (Might not be needed, depending on Action type number)
2. Determining the action to be performed and decide if a color is in the message
3. Send a request with the specified action to the server using the message format specified below
4. Wait for a response using a 1 second timeout period.
5. If a response arrives within the timeout period, print out the server response as shown in this document
6. If not, resend the message (same sequence number) for a maximum of 3 attempts before printing a ‘bulb not functioning’ error message.

The server will perform the following functions:

1. Read in 2 arguments from the command line:
2. IP address of server (127.0.0.1)
3. Port of server (e.g. 12001)
4. Initialize status variables (power and color to false and white, respectively)
5. Read in file named “supported-colors.txt” and store supported colors in database for use later
6. Listen for client requests until program is exited
7. If a request is received, decode the message and determine action to be performed
8. Return an error if the message is not formatted as specified or if color is not supported
9. Perform the action based on the action type number and send a response of the message format specified below
10. The program should still work if the supported-color file has colors added or removed from it

Test Cases:

All messages must adhere to the Message Format specified in this document:

1. Turn light bulb on and set color to a supported color
2. Change the light bulb to a different supported color
3. Change light bulb to an unsupported color
4. Check the status of the Light bulb
5. Turn the light off
6. Turn the light off while the light is already off

**Message Format**

In a action request, the application data has the following format in network byte order:

0 1 2 3 4 (bytes)

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

| Action Type Number (1) | Return Code (0) |

-----------------------------------------------------------------------------------

| Message Identifier(e.g. 654) |

-------------------------------------------------------------------------------------

| Color Length (e.g 3) | Power () |

------------------------------------------------------------------------------------

// Color String (e.g. ‘Red’) //

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

In a DNSresponse, the application data has the following format in network byte order:0 1 2 3 4 (bytes)

0 1 2 3 4 (bytes)

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

| Action Type Number (0) | Return Code (0) |

-----------------------------------------------------------------------------------

| Message Identifier(e.g. 654) |

-------------------------------------------------------------------------------------

| Color Length (e.g 3) | Power (e.g. true) |

------------------------------------------------------------------------------------

// Color String (e.g. ‘Red’) //

------------------------------------------------------------------------------------

// Response String (e.g. ‘OK’) //

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

Action Type Number (16 bits): 0, 1 2, 3, or 4. 0 on response, 1-4 on request. See map below to see what each number represents.

Return Code(16 bits): 0 on request; in response, 0 action was successful, 1 if something went wrong.

Message Identifier(32 bits): Uniquely identifies a message in a request, server echoes same number back in response. Should be generated randomly in range between 1 and 1000

Color Length(16 bits): In request and response, length of resource record in Color Section in bytes.

Power(16 bits): Empty in request. In response, boolean variable to signify if light is on (true) or off (false).

Color String(variable length): In request, String carrying the Color that is echoed back in the response.

Response string(variable length): In request, there is no response string. In response, the server returns a message that gives an overall report of the action. ‘OK’ for successful 1,2, and 4 action types. ‘Status: power = (on or off), color = (current color)’ for a successful 3 action type. For unsuccessful attempts, display an appropriate error message (see below for possible errors to account for).

**Action type number map**

The Action Type Numbers described earlier in this document will map to specific actions and the client and server should handle them appropriately. They are as follows:

1. Turn the light bulb on and set the color (**MUST HAVE** 4th parameter (Color) filled out)
2. Change the light color (**MUST HAVE** 4th parameter (Color) filled out)
3. Determine the status of the light: on/off, color (must **NOT HAVE** 4th parameter (Color) filled out)
4. Turn the light off (must **NOT HAVE** 4th parameter (Color) filled out)

**Errors to account for**

1. Bulb is not functioning (server is off or client is sending to the wrong address).
2. Color is not in the supported color list.
3. Message format is incorrect.

**Acceptable colors**

The file “supported-colors.txt” is simply a text file with a different color on each line. These colors must be parsed through on the server and stored in some type of database for use later. If the requested action color is among this database of colors, then it is accepted. Convert the request and database colors to lowercase before checking to avoid case sensitive issues.

**Test Output**

1.Test Case 1: Client Output Example(Turn lightbulb on and set color to supported color):

$ python3 iot-client.py 127.0.0.1 12001 1 Red

Sending Request to 127.0.0.1, 12001:

Action Type Number: 1

Message ID: 432

Color Length: 3 bytes

Color String: Red

Received Response from 127.0.0.1, 12001:

Return Code: 0

Action Type Number: 1

Message ID: 432

Color Length: 3 bytes

Power: true

Color String: Red

Response String: OK

Test Case 2: Client Output Example (Change color to new supported color)

$ python3 iot-client.py 127.0.0.1 12001 2 Green

Sending Request to 127.0.0.1, 12001:

Action Type Number: 2

Message ID: 129

Color Length: 5 bytes

Color String: Green

Received Response from 127.0.0.1, 12001:

Return Code: 0

Action Type Number: 2

Message ID: 129

Color Length: 5 bytes

Power: true

Color String: Green

Response String: OK

3.Test Case 3: Client Output Example (Change color to unsupported color)

$ python3 iot-client.py 127.0.0.1 12001 2 Pink

Sending Request to 127.0.0.1, 12001:

Action Type Number: 2

Message ID: 19

Color Length: 4 bytes

Color String: Pink

Received Response from 127.0.0.1, 12001:

Return Code: 1

Action Type Number: 2

Message ID: 19

Color Length: 4 bytes

Power: true

Color String: Pink

Response String: Color not Supported

4.Test Case 4: Client Output Example (Check status of lightbulb)

$ python3 iot-client.py 127.0.0.1 12001 3

Sending Request to 127.0.0.1, 12001:

Action Type Number: 3

Message ID: 977

Color Length: 0 bytes

Color String:

Received Response from 127.0.0.1, 12001:

Return Code: 0

Action Type Number: 3

Message ID: 977

Color Length: 0 bytes

Power: true

Color String:

Response String: Status = On, Color = Green

5.Test Case 5: Client Output Example (Turn the light off)

$ python3 iot-client.py 127.0.0.1 12001 4

Sending Request to 127.0.0.1, 12001:

Action Type Number: 4

Message ID: 389

Color Length: 0 bytes

Color String:

Received Response from 127.0.0.1, 12001:

Return Code: 0

Action Type Number: 4

Message ID: 389

Color Length: 0 bytes

Power: false

Color String:

Response String: OK

6.Test Case 6: Client Output Example (Check status of lightbulb while it is off)

$ python3 iot-client.py 127.0.0.1 12001 3

Sending Request to 127.0.0.1, 12001:

Action Type Number: 3

Message ID: 100

Color Length: 0 bytes

Power:

Color String:

Received Response from 127.0.0.1, 12001:

Return Code: 0

Action Type Number: 3

Message ID: 100

Color Length: 0 bytes

Power: false

Color String:

Response String: OK