Iot Protocol

Description: The protocol is based and build over UDP. to control IoT devices such as a light bulb.

Client Side Description:

The client will be responsible for the following functions:

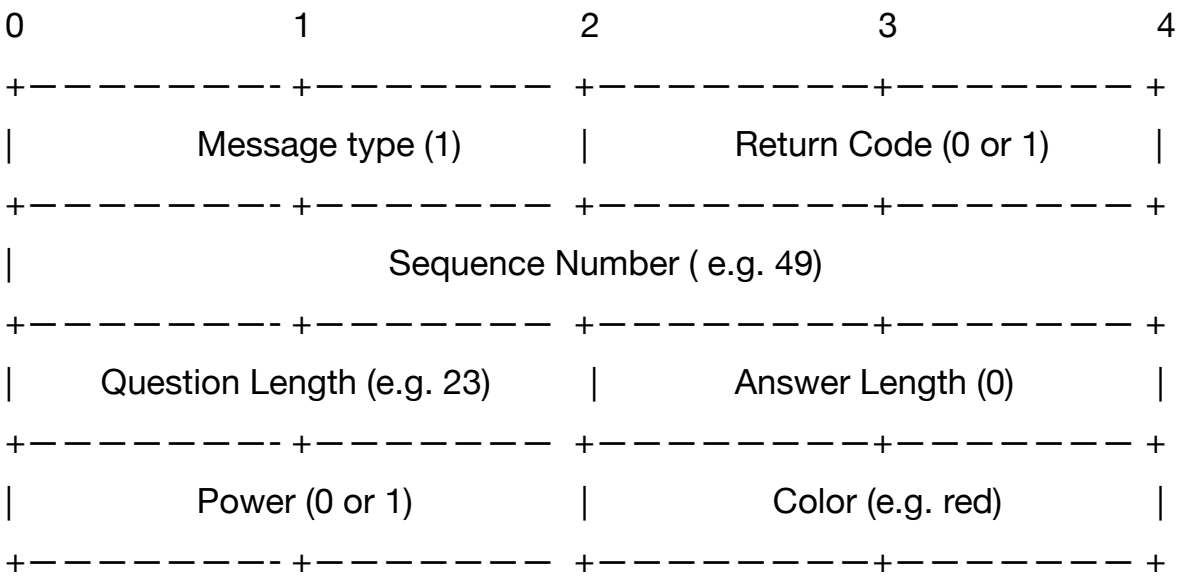
- A- Connect to the server by reading 4 arguments from the command line:
    1. IP address of the server (e.g. 127.0.0.1)
    2. Port of the server (e.g. 12000)
    3. Command (ON , OFF, SET, STATUS)
    4. Change bulb Color (accepted colors provided)
- B- Send request to turn on, turn off, change color and get status using the message format specified below.
- C- Wait for a response using 1 second timeout period:
    - If a response arrives within 1 second timeout period, print outs the server response with the status and other information specified in this document.
    - If no Response within the 1 second timeout period, Resend the message with the same request sequence number for a maximum of 3 attempts before printing an error message specified and exiting.

Server Side Description:

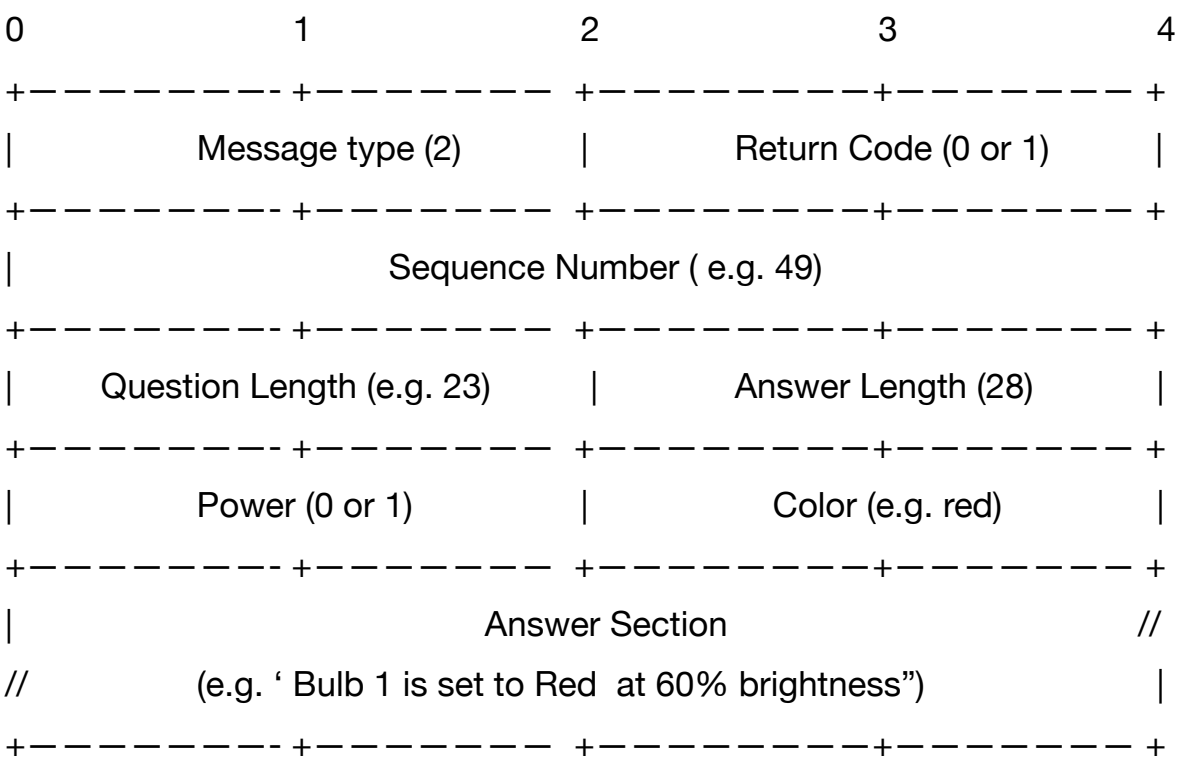
The server will be responsible for the following functions:

- A- Read in arguments from the command line:
    1. IP address (generally 127.0.0.1)
    2. Port of server (e.g. 12000)
- B- Read message and decode it to perform necessary functions from turning on the bulb, changing color, turn off the light .. and respond to client using message format specified below.
- C- Return error message if request message is incorrect (command or color missing or unsupported).

Message Format:



Response Format:



Details:

Message Type (16 bits): 1 on Request; 2 on response.

Return Code(16 bits): 0 if OK , 1 for unsupported Command error , 2 color not supported. 3 change color when OFF error, 4 for all other errors

Sequence Number(32 bits): Uniquely identify the request number and it can be randomly generated in range between 1 and 100

Question Length (16 bits): In request and response, length of command in Question Section in bytes.

Power(16 bits): 0 for Off; 1 for On.

Color (16 bits): list of accepted colors is provided. Supported colors : green, blue, red, yellow, white, black, aqua, purple.

Answer Length (16 bits): 0 in request (because no answer section). In response, length of color in bytes.

Extra Informations and assumptions:

Assume only one bulb exist and it is set by default to bulb number 1.

Turn off the light when OFF, and On when already ON , is not an error.

Keywords:

- *ON* : set the bulb status to on and color to white if no color is specified, otherwise the color specified color will be set.
- *OFF* : set the bulb status to OFF.
- *SET* : must be used with a color name otherwise no changes to the power or color
- <*color_name*> : color will set the bulb to ON and SET color to specified one.
- STATUS: return bulb status (power and color)

Test Output:

No Errors cases

1. Test Case 1: Client Output Example (turn on with default options):

$ python3 dns-client.py 127.0.0.1 12000 ON

```
Sending Request to 127.0.0.1, 12000:
Message ID: 23
Bulb No: 1
Power: ON
Color: default
Question Length: 18 Bytes
Answer Length: 0 Bytes


Received Response from 127.0.0.1, 12000:
return Code: 0 (No Errors)
Message ID: 23
Bulb No: 1
Power : ON
Color: white
Question Length: 18 Bytes
Question Length: 22 Bytes
Answer: ”Bulb 1 is ON and set to white.”
```

2. Test Case 2: Client Output Example (turn off the bulb):

$ python3 dns-client.py 127.0.0.1 12000 OFF

```
Sending Request to 127.0.0.1, 12000:
Message ID: 43
Bulb No: 1
Power: OFF
Color: default
Question Length: 23 Bytes
Answer Length: 0 Bytes

Received Response from 127.0.0.1, 12000:
return Code: 0 (No Errors)
Message ID: 43
Bulb No: 1
Power : OFF
```

```
Color:N/A
Question Length: 23 Bytes
Answer Length: 23 Bytes
Answer: ”Bulb 1 is OFF”
```

3. Test Case 3: Client Output Example (power on with a color):

$ python3 dns-client.py 127.0.0.1 12000 ON purple

```
Sending Request to 127.0.0.1, 12000:
Message ID: 19
Bulb No: 1
Power: OFF
Color: purple
Question Length: 23 Bytes
Answer Length: 0 Bytes

Received Response from 127.0.0.1, 12000:
return Code: 0 (No Errors)
Message ID: 19
Bulb No: 1
Power : ON
Color: purple
Question Length: 23 Bytes
Answer Length: 29 Bytes
Answer: ”Bulb 1 is ON and set to Purple”
```

4. Test Case 4: Client Output Example (get status):

$ python3 dns-client.py 127.0.0.1 12000 STATUS

```
Sending Request to 127.0.0.1, 12000:
Message ID: 19
Bulb No: 1
Power: default
Color: default
Question Length: 23 Bytes
Answer Length: 0 Bytes

Received Response from 127.0.0.1, 12000:
return Code: 0 (No Errors)
Message ID: 19
Bulb No: 1
Power : ON
Color: green
Question Length: 23 Bytes
Answer Length: 29 Bytes
Answer: ”Bulb 1 is ON and set too Green”
```

5. Test Case 5: Client Output Example (change color):

$ python3 dns-client.py 127.0.0.1 12000 SET aqua

```
Sending Request to 127.0.0.1, 12000:
Message ID: 19
Bulb No: 1
Power: default
Color: aqua
Question Length: 23 Bytes
Answer Length: 0 Bytes

Received Response from 127.0.0.1, 12000:
return Code: 0 (No Errors)
Message ID: 19
Bulb No: 1
Power : ON
Color: aqua
Question Length: 23 Bytes
Answer Length: 58 Bytes
Answer: ”Bulb 1 is ON and set to aqua”
```

ERRROR CASES

6. Test Case 6: Client Output Example (unknown command):

$ python3 dns-client.py 127.0.0.1 12000 change

```
Sending Request to 127.0.0.1, 12000:
Message ID: 19
Bulb No: 1
Power: default
Color: default
Question Length: 23 Bytes
Answer Length: 0 Bytes

Received Response from 127.0.0.1, 12000:
return Code: 1 (Unsupported Command)
Message ID: 19
Bulb No: 1
Power : unknown
Color: unknown
Question Length: 23 Bytes
Answer Length: 58 Bytes
Answer: ”Unknown command”
```

7. Test Case 7: Client Output Example (color not supported):

$ python3 dns-client.py 127.0.0.1 12000 SET pink

```
Sending Request to 127.0.0.1, 12000:
Message ID: 19
Bulb No: 1
Power: default
Color: Pink
Question Length: 23 Bytes
Answer Length: 0 Bytes

Received Response from 127.0.0.1, 12000:
return Code: 2 (Unsupported Color)
Message ID: 19
Bulb No: 1
Power : ON
Color: unknown
Question Length: 23 Bytes
Answer Length: 23 Bytes
Answer: ”Color Not Supported at the moment.”
```

8. Test Case 8: Client Output Example (using SET without a color):

$ python3 dns-client.py 127.0.0.1 12000 SET

```
Sending Request to 127.0.0.1, 12000:
Message ID: 19
Bulb No: 1
Power: default
Color: default
Question Length: 23 Bytes
Answer Length: 0 Bytes

Received Response from 127.0.0.1, 12000:
return Code: 3 (change color while OFF)
Message ID: 19
Bulb No: 1
Power : ON
Color: unknown
Question Length: 23 Bytes
Answer Length: 23 Bytes
Answer: ”Missing Parameter”
```

9. Test Case 8: Client Output Example (using SET with a color while OFF):

$ python3 dns-client.py 127.0.0.1 12000 SET red

```
Sending Request to 127.0.0.1, 12000:
Message ID: 19
Bulb No: 1
Power: default
Color: Red
Question Length: 23 Bytes
Answer Length: 0 Bytes
```

```
Received Response from 127.0.0.1, 12000:
return Code: 3 (change color while OFF)
Message ID: 19
Bulb No: 1
Power : OFF
Color: Red
Question Length: 23 Bytes
Answer Length: 23 Bytes
Answer: ”Please turn On the bulb before changing color”
```

10. Test Case 1: Client Output Example (Unreachable server):

$ python3 dns-client.py 127.0.0.1 12099 ON

```
Sending Request to 127.0.0.1, 12000:
Message ID: 19
Bulb No: 1
Power: ON
Color: default
Question Length: 23 Bytes
Answer Length: 0 Bytes


Request Timed out …
Sending Request to 127.0.0.1, 12000:
Request Timed out …
Sending Request to 127.0.0.1, 12000:
Request Timed out … Exiting Program.
```