Politechnika
Warszawska

Zaawansowane programowanie obiektowe i funkcyjne (Java)

Bramka REST API na TCP

Wojciech Gajda 304494

Krzysztof Leszek 313310

Marcin Latoszek 313307

Prowadzący: dr inż. Janusz Rafałko

Data oddania: **24 stycznia 2023**

Spis treści

1 Konspect

1.1 Motywacja

Rynek urządzeń IoT rozwija się niezwykle prężnie. Szczególnie widoczne jest to w przypadku inteligentnych urządzeń typu smart-home. Na rynku dostępne są coraz to nowsze urządzenia mające na celu ułatwić codzienne życie domownikom. Dzięki zwiększającej się różnorodności ceny urządzeń spadają, tym samym zwiększając swoją dostępność.

Jedną z gałęzi urządzeń smart-home stanowi inteligentne oświetlenie. Do niedawna liderem tej kartegorii była seria lamp i żarówek Hue, produkcji Philips. W ostatnim czasie na rynek wkroczyła firma Yeelight będąca spółką córką marki Xiaomi. Produkty z linii Yeelight okazują się być o rząd wielkości tańsze od swoich odpowiedników z serii Hue. Pozwala to myśleć o nich jako o doskonałej alternatywie. Jest jednak mały haczyk...

Rysunek 1: Żarówka Yeelight RGB wykorzystana w projekcie

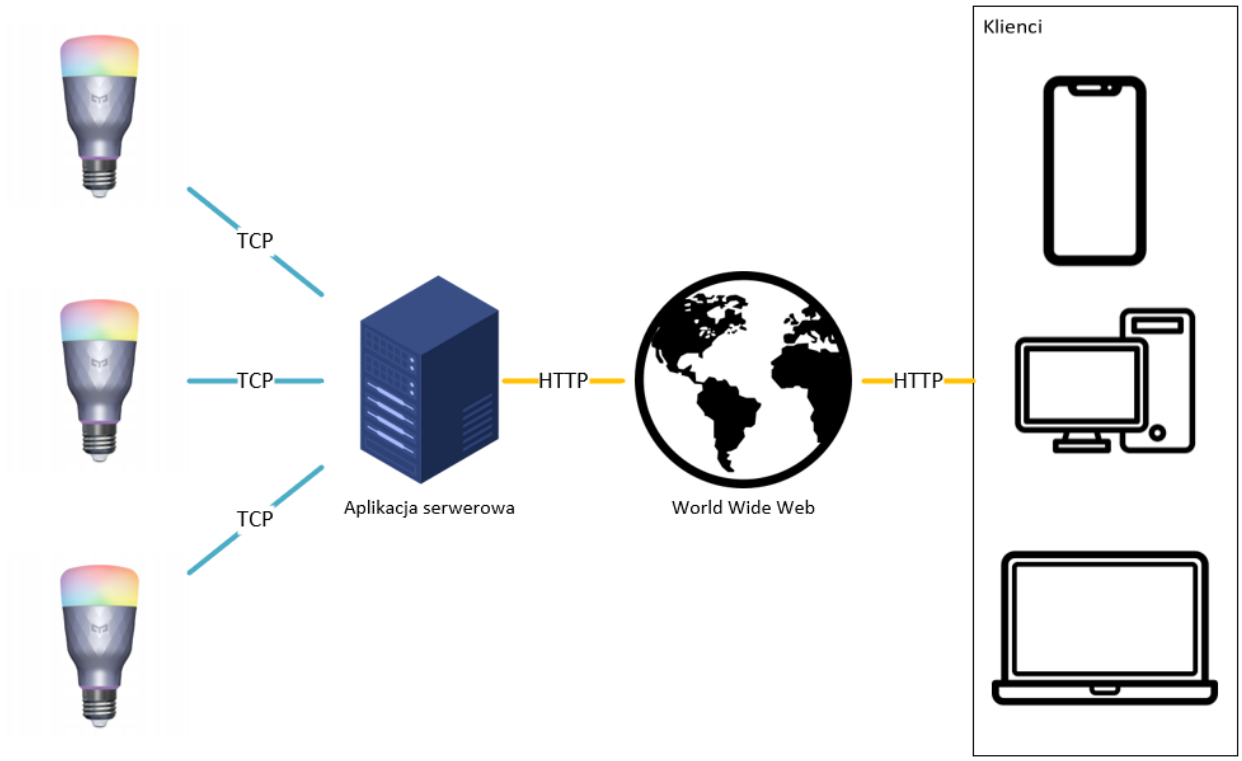
O ile w przypadku serii Hue komunikacją z urządzeniami odbywa się w sieci lokalnej, z ewentualnym wystawieniem dostępu na świat zewnętrzny, tak cała komunikacja z żarówkami Yeelight odbywa się przez serwer producenta zlokalizowany gdzieś na wschodniej części globu. Oprócz widocznych opóźnień, takie rozwiązanie posiada wiele wad, zaczynając od wytwarzania zbędnego ruchu sieciowego i kończąc na kwestiach związanych z bezpieczeństwem sieci. Obserwując tendencje na rynku wsparcie producenta może nie trwać długo, a ewentualne wyłączenie serwera głównego spowodowało by załamanie systemu. Aby zwiększyć porządną niezawodność systemu niezbędne jest wykluczenie zewnętrznych zależności.

Na szczęście, producent przewidział dla swoich urządzeń alternatywny sposób komunikacji. W dokumencie **Yeelight WiFi Light InterOperation Specification**[1] wyspecyfikowany został interfejs API pozwalający na wyszukiwanie żarówek za pośrednictwem protokołu UDP i komunikację z nimi przez protokół TCP.

W przypadku urządzeń IoT stanowość połączenia nie jest zjawiskiem porządanym. Komunikacja TCP nie jest powszechnie wspierana, gdyż wymaga ona przechowywania otwartych połączeń. Znacznie lepiej w tej roli sprawdza się komunikacja poprzez REST HTTP.

1.2 Cel projektu

Celem projektu jest stworzenie aplikacji serwerowej w postaci bramki wystawiającej API REST, która zarządza urządzeniami znajdującymi się w sieci lokalnej. Aplikacja znajduje wszystkie inteligentne urządzenia w sieci lokalnej, po czym nawiązuje z nimi połączenie. Urządzenie z którymi połączył się serwer stają się dostępne dla klientów łączących się protokołem HTTP. Serwer pełni jednocześnie role integratora. System okresowo odpytuje urządzenia w celu aktualizowania stania połączeń i ewentualnego usuwaniecia nieaktywnych urządzenia z listy dostępnych.



Rysunek 2: Struktura komiunikacji

[1] https://www.yeelight.com/download/Yeelight_Inter-Operation_Spec.pdf

1.3 Funkcjonalności projektu - use cases

Jako użytkownik mogę:

- pobrać listę dostępnych żarówek
- wyłączać i włączać żarówkę
- zmieniać kolor żarówki za pomocą modelu RGB żeby móc kontrolować kolor świecenia żarówki
- zmieniać jasność żarówki żeby kontrolować ilość dostarczanego przez żarówkę światła

Jako system mogę ...

- co określony czas przeszukiwać okolicę w celu odnalezienia żarówek oraz łączyć się z nimi żeby pozwolić użytkownikowi na ich kontrolę
- co określony czas mogę sprawdzić czy urządzenia w puli otwartych połączeń są nadal dostępne, aby usunąć urządzenia nieaktywne

1.4 Wybrane technologie

Aplikacja serwera zaprogramowana zostanie z wykorzystaniem języka Java 17. W aplikacji wykorzystany zostanie framework Spring Boot znacząco ułatwiający tworzenie serwera HTTP. Serwer zostanie stworzony z wykorzystaniem zasad Dependency Injection oraz Inversion of Control. Do komunikacji z urządzeniami wykorzystane zostaną standardowe biblioteki do komunikacji TCP/UDP. Do testów wykorzystany zostanie pakiet Mockito. Rozwiązane zostanie ponadto skonteneryzowane (Docker), a za platformę testową posłuży domowy serwer.

2 Przebieg projektu

2.1 Funcjonalności w finalnej wersji projektu

Aplikacja...

- po uruchomieniu wykorzystuje UDP do skanowania sieci w poszukiwaniu żarówek. Wykrywane są żarówki w zasięgu broadcastu UDP (na ogół oznacza to sieć lokalną)
- skanowanie powtarzanie jest w cron, co określony interwał czasu
- aplikacja nasłuchuje zmian stanu podłączonych żarówek, nawet jeśli pochodzą one z innych źródeł.

Jako użytkownik mogę ...

- pobrać listę żarówek lub informację o konkretnej żarówce
- włączyć i wyłączyć żarówkę
- zmienić kolor żarówki (RGB lub HSV)
- zmienić barwę temperaturową żarówki
- zmienić jasność żarówki
- zmienić nazwę żarówki
- rozpocząć ponowne skanowanie sieci
- rozpocząć migotanie żarówki w rytm muzyki.

2.2 Algorytm dobory koloru i jasności żarówki do dźwięku

Migotanie żarówki w rytm muzyki polega na takim dobraniu barwy i jasności światła aby w połączeniu z odtwarzaną muzyką dawało to wrażenie spójności. W celu rozpoznania charakteru odtwarzanego dźwięku jest on dzielony na mniejsze próbki. Każda próbka zawiera 8192 wartości co przy częstotliwości próbkowania WAV wynoszącej 44.1kHz oznacza ze próbka zawiera przedział ok. $\frac{8192}{44100} = 18,6ms$. Dla tak pobranego wektora danych przeprowadzana jest Szybka Dyskretna Transformata Fouriera mająca na celu ujawnić częstotliwości dominujące w sygnale.

Powyższa transformata każdej częstotliwości z zakresu przyporządkowuje liczbę zespoloną. Moduł tej liczby odpowiada amplitudzie składowej, zaś jej argument odpowiada przesunięciu w fazie. Pierwszym pomysłem na wybór koloru i jasności jest wybór częstotliwość o maksymalnym module. Wtedy moduł częstotliwości skalowałby się na jasność, a częstotliwość można by przeskalować na barwę np. w modelu HSV. Rozwiązanie to okazało się jednak nieskuteczne. Z obserwacji wynika, że w większości odtwarzanych dźwięków dominująca jest zwykle jedna częstotliwość z zakresu 300-1000Hz. Powoduje to małą zmienność w barwie w trakcie działania układu.

W związku z powyższym należało poszukać innego rozwiązania. Inspiracją stało się klasyczne podejście wykorzystywane w układach elektronicznych wbudowanych w kolumny gło-

snikowe. W kolumnach z dodatkowymi efektami iluminacyjnymi wykorzystuje się następujące rozwiązanie. Do linii sygnałowej podłączane zostają 3 filtry: dolno-, środkowo- i górnoprzepustowy przypuszczające odpowiednio niskie, średnie i wysokie tony. do wyjścia z każdego z filtrów podłączone są reflektory.

Inspirując się tym rozwiązaniem algorytm dzieli wynik FFT i sumuje w przedziałach 0-300Hz, 300-3000Hz, +3000Hz amplitudy poszczególnych częstotliwości. Następnie sumy te przekładają reprezentują kolejno barwy czerwoną, zieloną i niebieską. Jasność natomiast zależna jest od średniej tych sum.

2.3 DTO

2.3.1 VersionDTO

Obiekt VersionDTO to DTO przechowujące akutalną wersję kodu. Pobranie go możlwie jest w endpoint'cie /version

Przykładowy JSON:

```
{
  "version": "v1.1"
}
```

2.3.2 Bulb

Obiekt Bulb to DTO przechowujące wszystkie informacje o żarówce.
Przykładowy JSON:


```
{
  "location": {
    "address": {
      "canonicalHostName": "string",
      "hostAddress": "string",
      "address": [
        "string"
      ],
      "hostName": "string",
      "linkLocalAddress": true,
      "multicastAddress": true,
      "anyLocalAddress": true,
      "loopbackAddress": true,
      "siteLocalAddress": true,
      "mcglobal": true,
      "mcnodeLocal": true,
      "mclinkLocal": true,
      "mcsiteLocal": true,
      "mcorgLocal": true
    },
    "port": 0,
    "unresolved": true,
    "hostName": "string",
    "hostString": "string"
  },
  "serial": "string",
  "model": "string",
  "support": [
    "string"
  ],
  "power": true,
  "bright": 0,
  "colorMode": 0,
  "ct": 0,
  "rgb": 0,
  "hue": 0,
  "saturation": 0,
  "name": "string"
}
```

2.4 REST API

Metoda: PUT, ścieżka: /{id}
Endpoint umożliwia włączenie lub wyłączenie żarówki.

Parametry:
- id (string) - id żarówki, wymagany, path
- power (boolean) - czy żarówka ma zostać włączona, wymagany, query
- changingTimeInMillis (integer) - czas trwania operacji, query

Request body: *brak*

Odpowiedzi:
- Kod 200 - Poprawnie włączono / wyłączono żarówkę (application/json)

Metoda: PUT, ścieżka: /{id}/rgb
Endpoint umożliwia na zmianę koloru żarówki za pomocą modelu rgb.

Parametry:
- id (string) - id żarówki, wymagany, path
- red (integer) - składowa r modelu rgb, wymagany, query
- green (integer) - składowa g modelu rgb, wymagany, query
- blue (integer) - składowa b modelu rgb, wymagany, query
- changingTimeInMillis (integer) - czas trwania operacji, query

Request body: *brak*

Odpowiedzi:
- Kod 200 - Poprawnie zmieniono kolor żarówki (application/json)

Metoda: PUT, ścieżka: /{id}/hsv
Endpoint umożliwia na zmianę koloru żarówki za pomocą modelu hsv.

Parametry:
- id (string) - id żarówki, wymagany, path
- hue (integer) - składowa hue modelu hsv, wymagany, query
- sat - składowa saturation modelu hsv, wymagany, query
- changingTimeInMillis (integer) - czas trwania operacji, query

Request body: *brak*

Odpowiedzi:
- Kod 200 - Poprawnie zmieniono kolor żarówki (application/json)

Metoda: PUT, ścieżka: /{id}/fft
Endpoint umożliwia na migotanie żarówki w rytm muzyki (pliku .wav).

Parametry:
- id (string) - id żarówki, wymagany, path
- uri (string) - link sieciowy z dostępem do pliku .wav, wymagany, query

Request body: *brak*

Odpowiedzi:
- Kod 200 - Poprawnie zmieniono tryb świecenia żarówki na migoczący się w rytm muzyki (application/json)

**Metoda: PUT, ścieżka: /{id}/ct$_a$*bx*
Endpoint umożliwia na zmianę barwę temperaturową żarówki.

Parametry:
- id (string) - id żarówki, wymagany, path
- ct_value (integer) - temperatura świecenia żarówki, wymagany, query
- changingTimeInMillis (integer) - czas trwania operacji, query

Request body: *brak*

Odpowiedzi:
- Kod 200 - Poprawnie zmieniono barwę temperaturową żarówki (application/json)

Metoda: PUT, ścieżka: /{id}/brightness
Endpoint umożliwia zmianę jasności swiecenia żarówki.

Parametry:
- id (string) - id żarówki, wymagany, path
- brightness (byte) - nowa jasność żarówki, wymagany, query
- changingTimeinMillis(integer) - czas trwania operacji, query

Request body: *brak*

Odpowiedzi:
- Kod 200 - Poprawnie zmieniono jasność żarówki (application/json)

Metoda: GET, ścieżka: /bulbs/{id}
Endpoint umożliwia pobranie informacji na temat żarówki o danym id.

Parametry:
- id (string) - id żarówki, wymagany, path

Request body: *brak*

Odpowiedzi:
- Kod 200 - Poprawnie pobrano, odpowiedź zawiera informacje o żarówce (application/json)

Metoda: GET, ścieżka: /bulbs/
Endpoint umożliwia pobranie listy wszystkich żarówek

Parametry: *brak*
Request body: *brak*

Odpowiedzi:
- Kod 200 - Poprawnie pobrano, odpowiedź zawiera listę żarówek wraz z informacjami o nich (application/json)

Metoda: PUT, ścieżka: /bulbs/{id}
Endpoint umożliwia nadanie własnej nazwy żarówce.

Parametry:
- id (string) - id żarówki, wymagany, path
- name (string) - nowa nazwa żarówki, wymagany, query

Request body: *brak*

Odpowiedzi:
- Kod 200 - Poprawnie nadano nazwę żarówce (application/json)

Metoda: GET, ścieżka: /version
Podstawowy endpoint, umożliwia sprawdzenie wersji

Parametry: *brak*

Request body: *brak*

Odpowiedzi:
- Kod 200 - Odpowiedź zawiera numer wersji aplikacji

Metoda: GET, ścieżka: /forcescan
Endpoint umożliwia wymuszenie szukania dostępnych urządzeń.

Parametry: *brak*

Request body: *brak*

Odpowiedzi:

- Kod 200 - Przeprowadzono wymuszony skan

2.5 Specyfikacja API w standardzie OpenAPI 3.0


```
{
  "openapi": "3.0.1",
  "info": {
    "title": "OpenAPI definition",
    "version": "v0"
  },
  "servers": [
    {
      "url": "http://xevix.tplinkdns.com:6060",
      "description": "Generated server url"
    }
  ],
  "paths": {
    "/{id}": {
      "put": {
        "tags": [
          "Power"
        ],
        "operationId": "getBulb",
        "parameters": [
          {
            "name": "id",
            "in": "path",
            "required": true,
            "schema": {
              "type": "string"
            }
          },
          {
            "name": "power",
            "in": "query",
            "required": true,
            "schema": {
              "type": "boolean"
            }
          },
          {
            "name": "changingTimeInMillis",
            "in": "query",
            "required": false,
            "schema": {
              "type": "integer",
              "format": "int32"
```

```
        }
      }
    ],
    "responses": {
      "200": {
        "description": "OK",
        "content": {
          "*/*": {
            "schema": {
              "type": "string"
            }
          }
        }
      }
    }
  }
},
"/{id}/rgb": {
  "put": {
    "tags": [
      "RGB"
    ],
    "operationId": "setBulbRgb",
    "parameters": [
      {
        "name": "id",
        "in": "path",
        "required": true,
        "schema": {
          "type": "string"
        }
      },
      {
        "name": "red",
        "in": "query",
        "required": true,
        "schema": {
          "maximum": 255,
          "minimum": 0,
          "type": "integer",
          "format": "int32"
        }
      },
      {
        "name": "green",
```

```
          "in": "query",
          "required": true,
          "schema": {
            "maximum": 255,
            "minimum": 0,
            "type": "integer",
            "format": "int32"
          }
        },
        {
          "name": "blue",
          "in": "query",
          "required": true,
          "schema": {
            "maximum": 255,
            "minimum": 0,
            "type": "integer",
            "format": "int32"
          }
        },
        {
          "name": "changingTimeInMillis",
          "in": "query",
          "required": false,
          "schema": {
            "type": "integer",
            "format": "int32"
          }
        }
      ],
      "responses": {
        "200": {
          "description": "OK",
          "content": {
            "*/*": {
              "schema": {
                "type": "string"
              }
            }
          }
        }
      }
    }
  },
  "/{id}/hsv": {
```

```
"put": {
  "tags": [
    "HSV"
  ],
  "operationId": "setBulbHsv",
  "parameters": [
    {
      "name": "id",
      "in": "path",
      "required": true,
      "schema": {
        "type": "string"
      }
    },
    {
      "name": "hue",
      "in": "query",
      "required": true,
      "schema": {
        "maximum": 359,
        "minimum": 0,
        "type": "integer",
        "format": "int32"
      }
    },
    {
      "name": "sat",
      "in": "query",
      "required": true,
      "schema": {
        "maximum": 100,
        "minimum": 0,
        "type": "integer",
        "format": "int32"
      }
    },
    {
      "name": "changingTimeInMillis",
      "in": "query",
      "required": false,
      "schema": {
        "type": "integer",
        "format": "int32"
      }
    }
```

```
    ],
    "responses": {
      "200": {
        "description": "OK",
        "content": {
          "*/*": {
            "schema": {
              "type": "string"
            }
          }
        }
      }
    }
  }
},
"/{id}/fft": {
  "put": {
    "tags": [
      "FFT"
    ],
    "operationId": "setBulbHsv_1",
    "parameters": [
      {
        "name": "id",
        "in": "path",
        "required": true,
        "schema": {
          "type": "string"
        }
      },
      {
        "name": "uri",
        "in": "query",
        "required": true,
        "schema": {
          "type": "string"
        }
      }
    ],
    "responses": {
      "200": {
        "description": "OK",
        "content": {
          "*/*": {
            "schema": {
```

```
                "type": "string"
              }
            }
          }
        }
      }
    }
  },
  "/{id}/ct_abx": {
    "put": {
      "tags": [
        "Light temperature"
      ],
      "operationId": "setBulbRgb_1",
      "parameters": [
        {
          "name": "id",
          "in": "path",
          "required": true,
          "schema": {
            "type": "string"
          }
        },
        {
          "name": "ct_value",
          "in": "query",
          "required": true,
          "schema": {
            "maximum": 6500,
            "minimum": 1700,
            "type": "integer",
            "format": "int32"
          }
        },
        {
          "name": "changingTimeInMillis",
          "in": "query",
          "required": false,
          "schema": {
            "type": "integer",
            "format": "int32"
          }
        }
      ],
      "responses": {
```

```
      "200": {
        "description": "OK",
        "content": {
          "*/*": {
            "schema": {
              "type": "string"
            }
          }
        }
      }
    }
  },
  "/{id}/brightness": {
    "put": {
      "tags": [
        "Brightness"
      ],
      "operationId": "setBrightness",
      "parameters": [
        {
          "name": "id",
          "in": "path",
          "required": true,
          "schema": {
            "type": "string"
          }
        },
        {
          "name": "brightness",
          "in": "query",
          "required": true,
          "schema": {
            "maximum": 100,
            "minimum": 1,
            "type": "string",
            "format": "byte"
          }
        },
        {
          "name": "changingTimeInMillis",
          "in": "query",
          "required": false,
          "schema": {
            "type": "integer",
```

```
            "format": "int32"
          }
        }
      ],
      "responses": {
        "200": {
          "description": "OK",
          "content": {
            "*/*": {
              "schema": {
                "type": "string"
              }
            }
          }
        }
      }
    }
  },
  "/bulbs/{id}": {
    "get": {
      "tags": [
        "Bulb"
      ],
      "operationId": "getBulb_1",
      "parameters": [
        {
          "name": "id",
          "in": "path",
          "required": true,
          "schema": {
            "type": "string"
          }
        }
      ],
      "responses": {
        "200": {
          "description": "OK",
          "content": {
            "*/*": {
              "schema": {
                "$ref": "#/components/schemas/Bulb"
              }
            }
          }
        }
      }
```

```
      }
    },
    "put": {
      "tags": [
        "Name"
      ],
      "operationId": "setName",
      "parameters": [
        {
          "name": "id",
          "in": "path",
          "required": true,
          "schema": {
            "type": "string"
          }
        },
        {
          "name": "name",
          "in": "query",
          "required": true,
          "schema": {
            "type": "string"
          }
        }
      ],
      "responses": {
        "200": {
          "description": "OK",
          "content": {
            "*/*": {
              "schema": {
                "type": "string"
              }
            }
          }
        }
      }
    }
  },
  "/version": {
    "get": {
      "tags": [
        "Common"
      ],
      "operationId": "api",
```

```
    "responses": {
      "200": {
        "description": "OK",
        "content": {
          "*/*": {
            "schema": {
              "$ref": "#/components/schemas/VersionDTO"
            }
          }
        }
      }
    }
  }
},
"/forceScan": {
  "get": {
    "tags": [
      "Common"
    ],
    "operationId": "searchForBulbs",
    "responses": {
      "200": {
        "description": "OK",
        "content": {
          "*/*": {
            "schema": {
              "type": "string"
            }
          }
        }
      }
    }
  }
},
"/bulbs": {
  "get": {
    "tags": [
      "Bulb"
    ],
    "operationId": "getBulbs",
    "responses": {
      "200": {
        "description": "OK",
        "content": {
          "*/*": {
```

```
                "schema": {
                  "type": "array",
                  "items": {
                    "$ref": "#/components/schemas/Bulb"
                  }
                }
              }
            }
          }
        }
      }
    }
  },
  "components": {
    "schemas": {
      "VersionDTO": {
        "type": "object",
        "properties": {
          "version": {
            "type": "string"
          }
        },
        "description": "VersionDTO"
      },
      "Bulb": {
        "type": "object",
        "properties": {
          "location": {
            "type": "object",
            "properties": {
              "address": {
                "type": "object",
                "properties": {
                  "canonicalHostName": {
                    "type": "string"
                  },
                  "hostAddress": {
                    "type": "string"
                  },
                  "address": {
                    "type": "array",
                    "items": {
                      "type": "string",
                      "format": "byte"
                    }
```

```
      },
      "hostName": {
        "type": "string"
      },
      "linkLocalAddress": {
        "type": "boolean"
      },
      "multicastAddress": {
        "type": "boolean"
      },
      "anyLocalAddress": {
        "type": "boolean"
      },
      "loopbackAddress": {
        "type": "boolean"
      },
      "siteLocalAddress": {
        "type": "boolean"
      },
      "mcglobal": {
        "type": "boolean"
      },
      "mcnodeLocal": {
        "type": "boolean"
      },
      "mclinkLocal": {
        "type": "boolean"
      },
      "mcsiteLocal": {
        "type": "boolean"
      },
      "mcorgLocal": {
        "type": "boolean"
      }
    }
  },
  "port": {
    "type": "integer",
    "format": "int32"
  },
  "unresolved": {
    "type": "boolean"
  },
  "hostName": {
    "type": "string"
```

```
      },
      "hostString": {
        "type": "string"
      }
    }
  },
  "serial": {
    "type": "string"
  },
  "model": {
    "type": "string"
  },
  "support": {
    "type": "array",
    "items": {
      "type": "string"
    }
  },
  "power": {
    "type": "boolean"
  },
  "bright": {
    "type": "integer",
    "format": "int32"
  },
  "colorMode": {
    "type": "integer",
    "format": "int32"
  },
  "ct": {
    "type": "integer",
    "format": "int32"
  },
  "rgb": {
    "type": "integer",
    "format": "int32"
  },
  "hue": {
    "type": "integer",
    "format": "int32"
  },
  "saturation": {
    "type": "integer",
    "format": "int32"
  },
```

```
            "name": {
              "type": "string"
            }
          }
        }
      }
    }
}
```