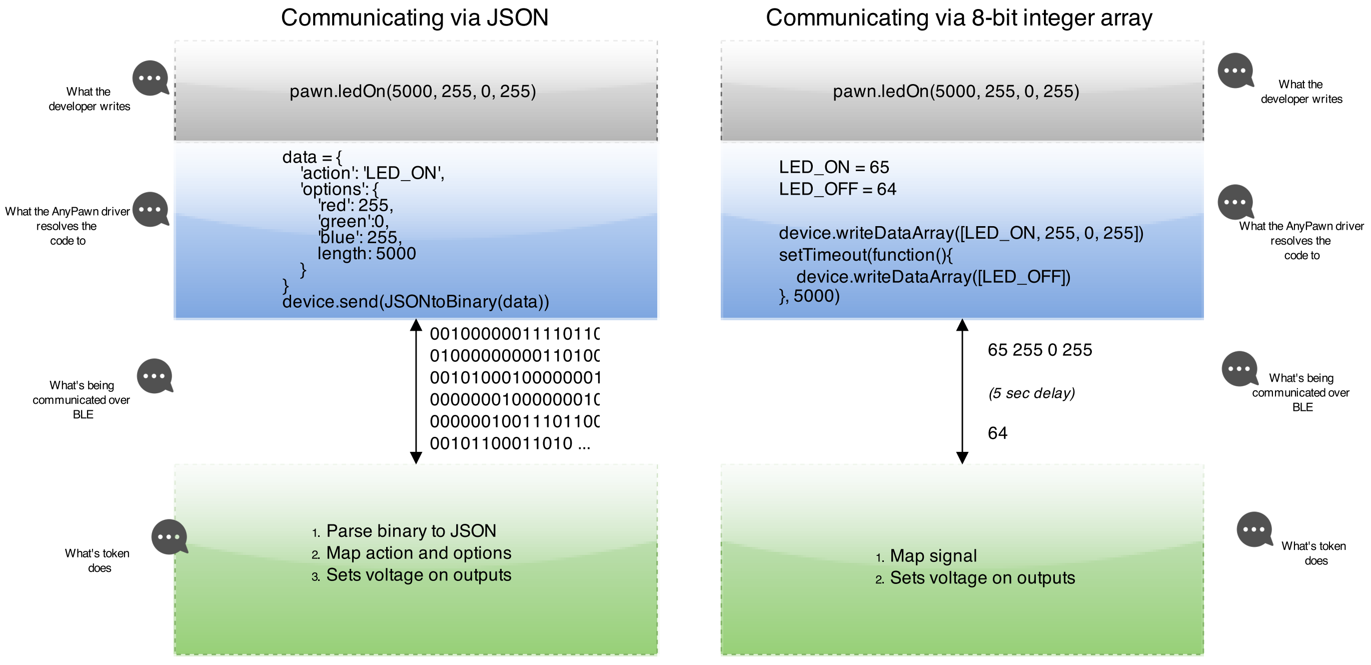

Communicating via JSON

What the developer writes

```
pawn.ledOn(5000, 255, 0, 255)
```

What the AnyPawn driver resolves the code to

```
data = {
   'action': 'LED_ON',
   'options': {
      'red': 255,
      'green':0,
      'blue': 255,
      length: 5000
   }
}
device.send(JSONtoBinary(data))
```

What's being communicated over BLE

```
001000000011110110
010000000001101000
001010001000000001
000000010000000010
000000100111011000
00101100011010 ...
```

What's token does

1. Parse binary to JSON
2. Map action and options
3. Sets voltage on outputs

Communicating via 8-bit integer array

What the developer writes

```
pawn.ledOn(5000, 255, 0, 255)
```

What the AnyPawn driver resolves the code to

```
LED_ON = 65
LED_OFF = 64

device.writeDataArray([LED_ON, 255, 0, 255])
setTimeout(function(){
   device.writeDataArray([LED_OFF])
}, 5000)
```

What's being communicated over BLE

```
65 255 0 255

(5 sec delay)

64
```

What's token does

1. Map signal
2. Sets voltage on outputs

+ Hardware interface more independant from driver

+ Parsing potentially faster on token
+ Smaller BLE payload
+ Less pawn battery usage (unknown if notable amount)
+ Some functional extendability without changing firmware
- Less semantic token interface if used without driver