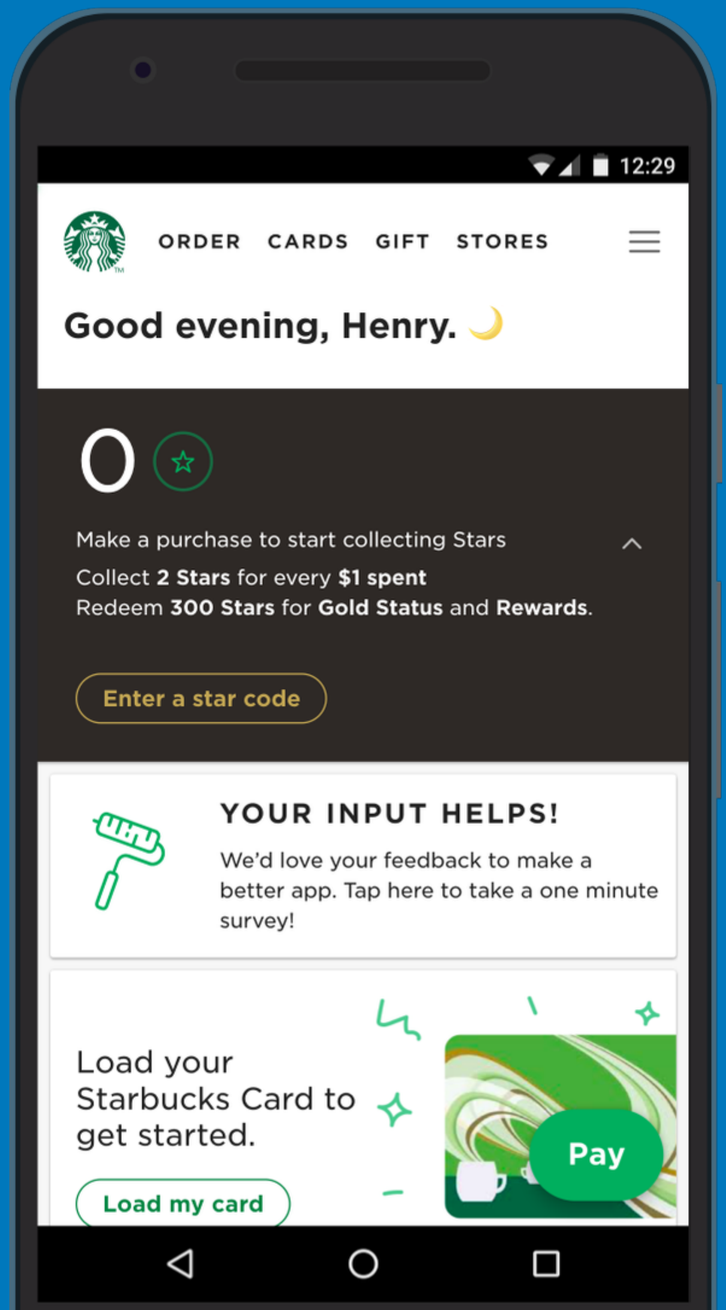
# *fun with bluetooth*

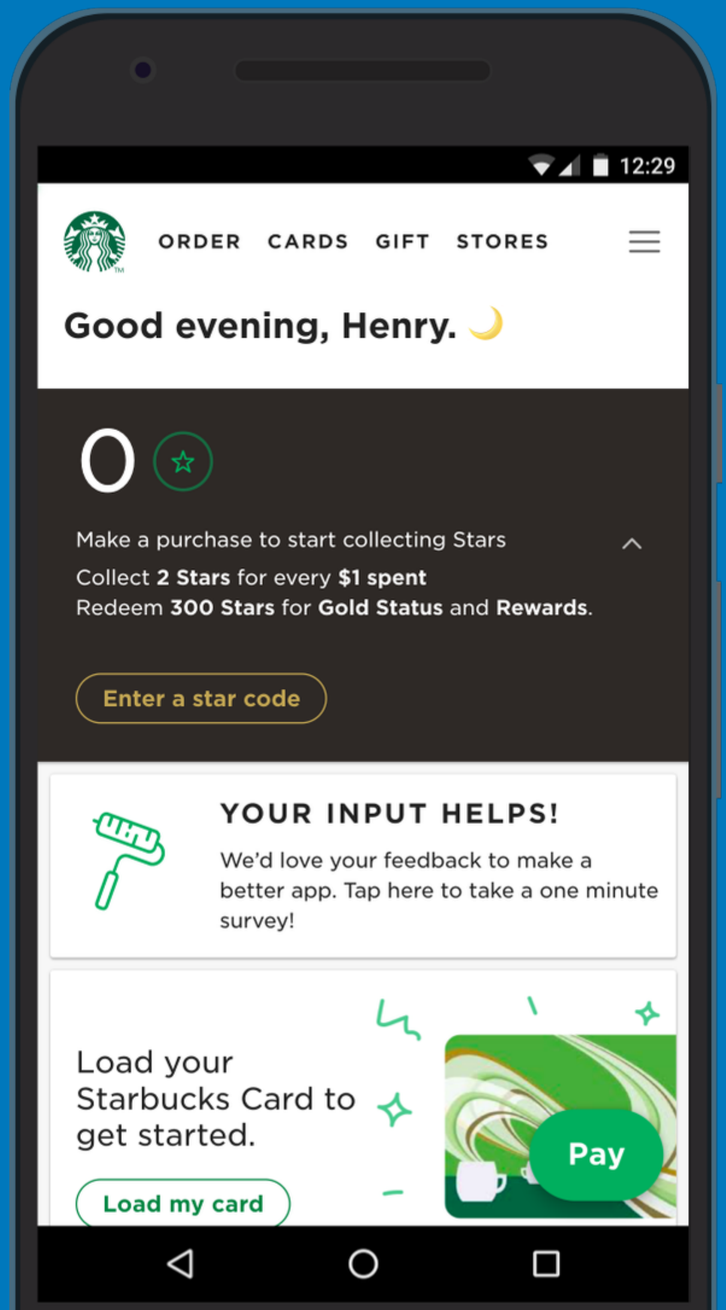
*why?*


# *progressive web apps*

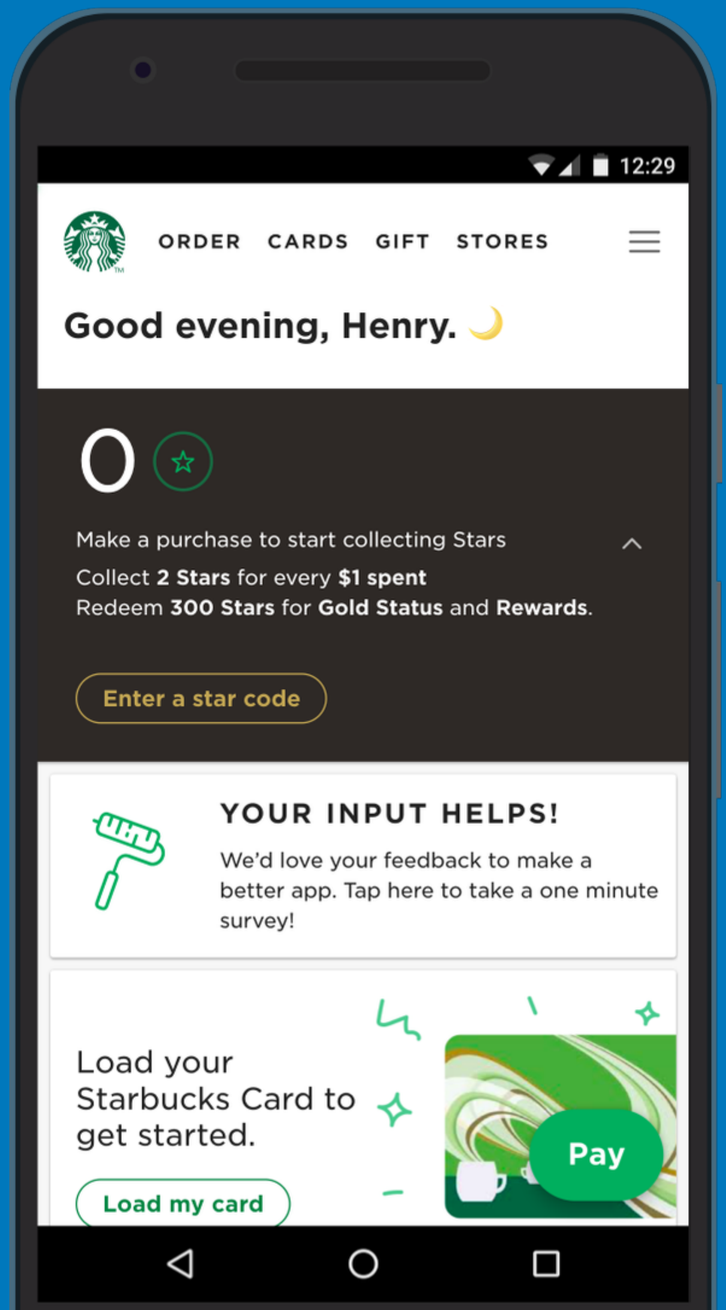
*pwa's are great!*

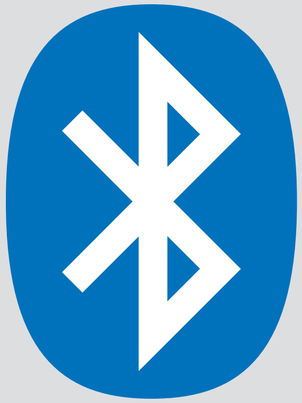


*but...*

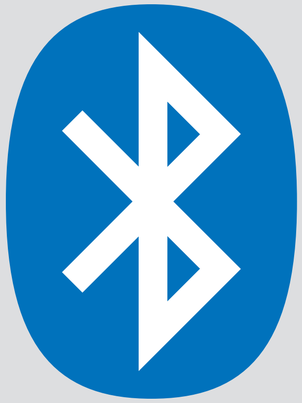

12:29
ORDER CARDS GIFT STORES
Good evening, Henry.
0
Make a purchase to start collecting Stars
Collect 2 Stars for every $1 spent
Redeem 300 Stars for Gold Status and Rewards.
Enter a star code
YOUR INPUT HELPS!
We'd love your feedback to make a better app. Tap here to take a one minute survey!
Load your Starbucks Card to get started.
Load my card
Pay

*but...*


12:29
ORDER CARDS GIFT STORES
Good evening, Henry.
0
Make a purchase to start collecting Stars
Collect 2 Stars for every $1 spent
Redeem 300 Stars for Gold Status and Rewards.
Enter a star code
YOUR INPUT HELPS!
We'd love your feedback to make a better app. Tap here to take a one minute survey!
Load your Starbucks Card to get started.
Load my card
Pay



bluetooth



bluetooth sucks

classic bluetooth

the reason everybody hates bluetooth

*vs.*

control drones and other cool shit

bluetooth low energy

# bluetooth low energy

also known as

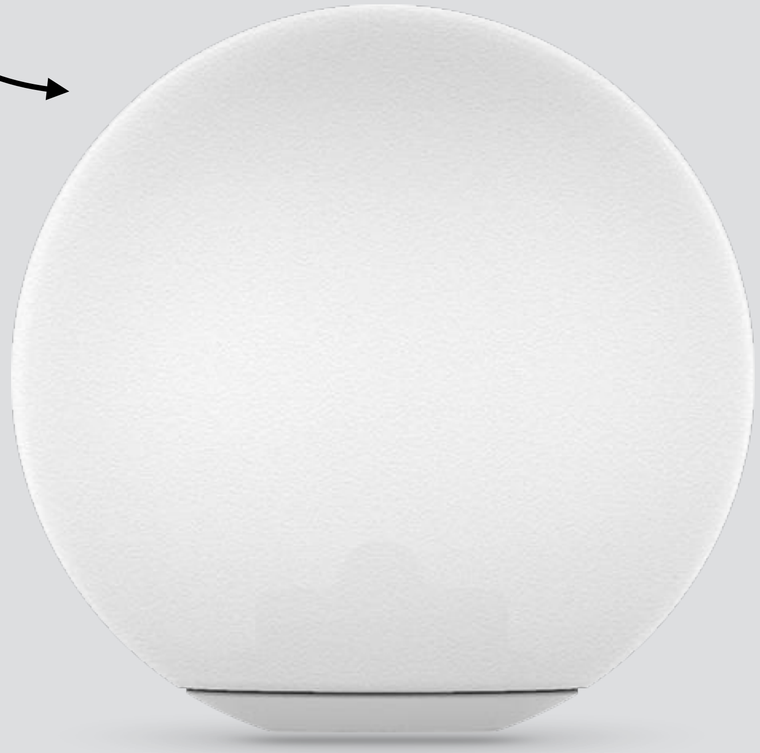
Bluetooth Smart

Bluetooth LE

BLE

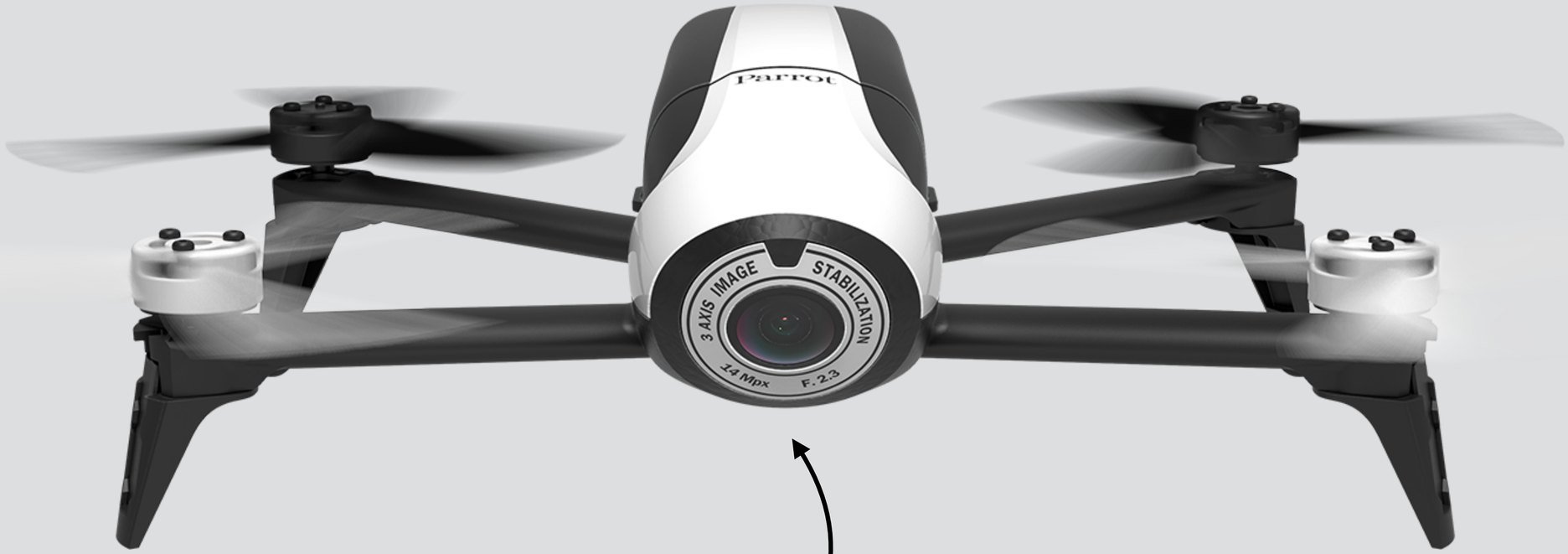
Bluetooth 4

playbulb sphere




playbulb

spherio bb-8




parrot mini drone


activity tracker

*the boring theoretical stuff*


central


peripheral




central

# generic attribute profile

generic attribute profile ?

generic attribute profile

gatt, because gap was already taken



~~central~~

client

~~peripheral~~

server


device information

light

multiple services per device


device information

battery

flight control

device information

battery

steering control

- device information
- battery
- heart rate

dev inf on

b y

heart rate


device information

battery

heart rate

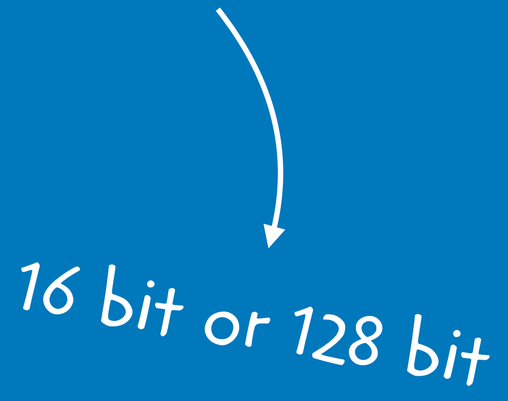
## device information

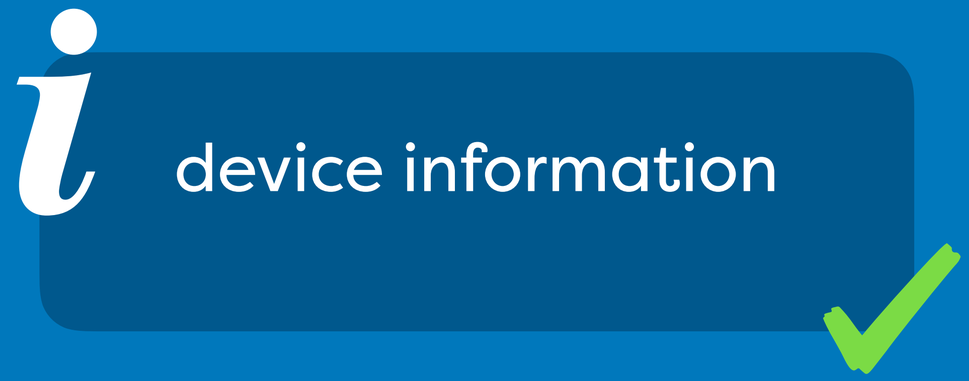
manufacturer
model number
serial number
hardware revision
firmware revision
software revision
...

multiple characteristics per service

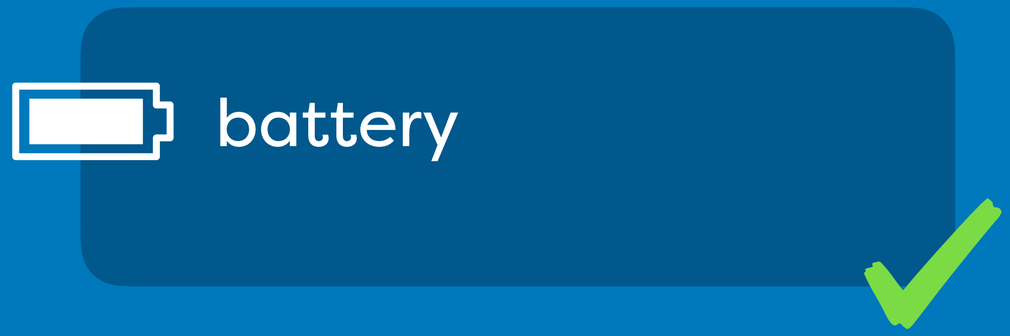
| | |
|---|---|
| server | array of objects |
| service | object |
| characteristic | property |
| value | value |

services and characteristics are identified by uuid's






| | |
|---|---|
| 16 bit | 0x180A |
| 128 bit | 0000180A-0000-1000-8000-00805F9B34FB |




16 bit `0x180F`

128 bit `0000180F-0000-1000-8000-00805F9B34FB`

light

steering control

flight c

still, everybody does this

16 bit `not recommended`

128 bit `any UUID outside of the range`

`xxxxxxxx-0000-1000-8000-00805F9B34FB`

## device information

manufacturer
model number
serial number
hardware revision
firmware revision
software revision
...


bad for readability,
good for saving bandwidth

each characteristic supports one or more of these

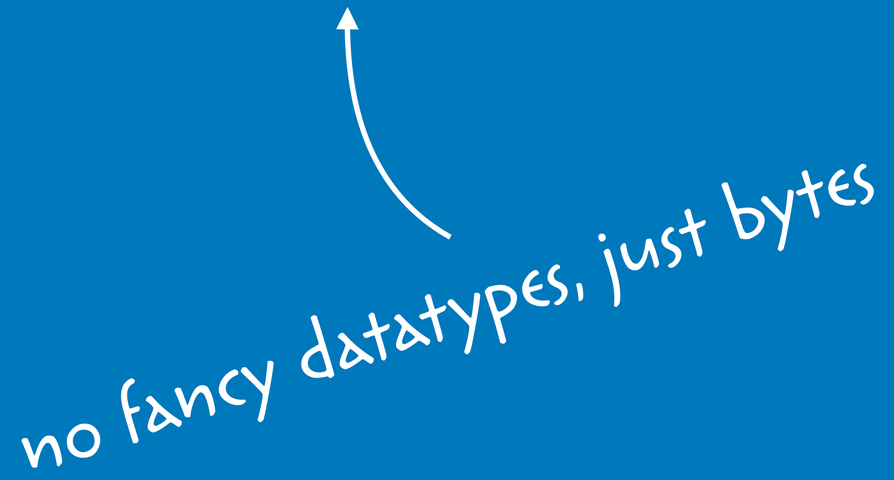
read

write

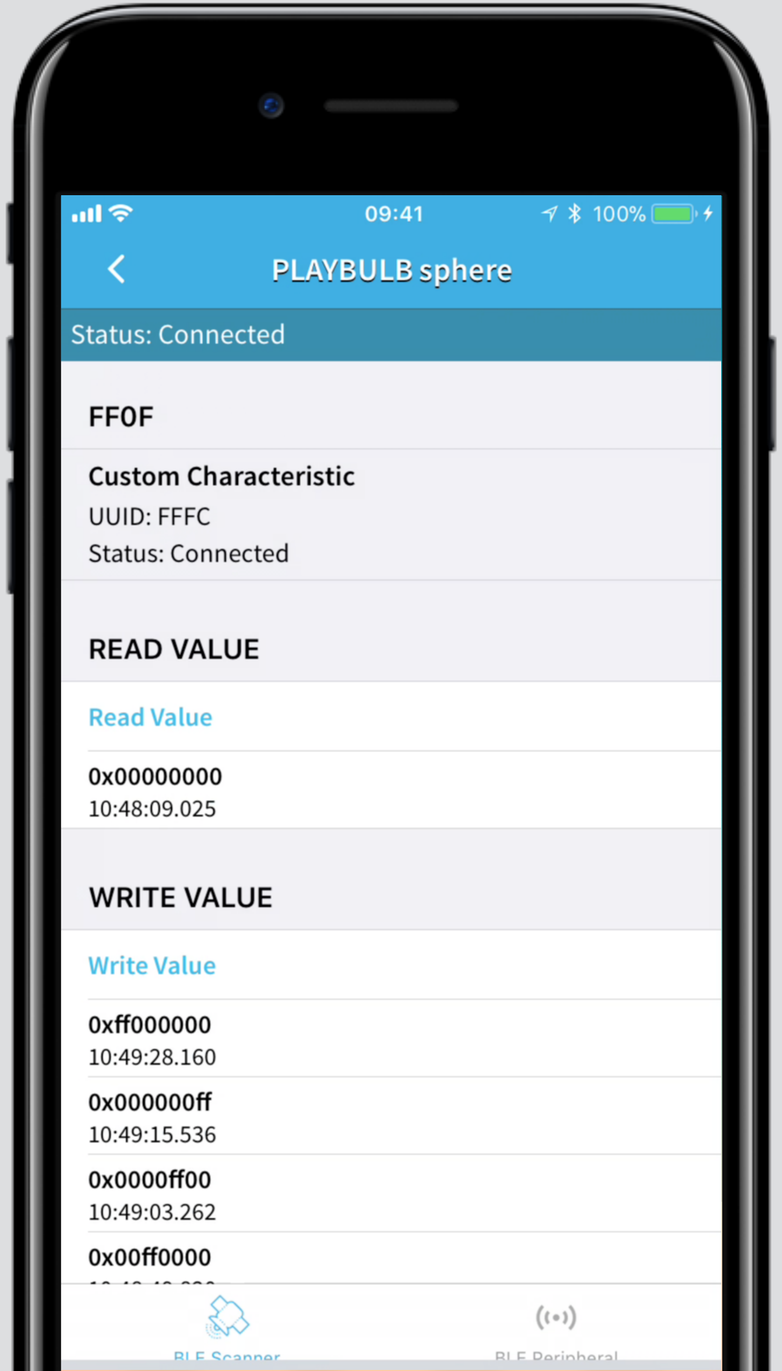
write without response

notify

every value is an array of bytes

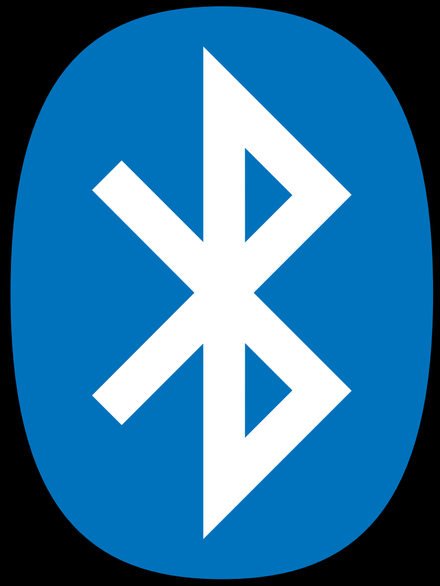


*pfew...*


09:41
100%
PLAYBULB sphere
Status: Connected
FF0F
Custom Characteristic
UUID: FFFC
Status: Connected
READ VALUE
Read Value
0x00000000
10:48:09.025
WRITE VALUE
Write Value
0xff000000
10:49:28.160
0x000000ff
10:49:15.536
0x0000ff00
10:49:03.262
0x00ff0000

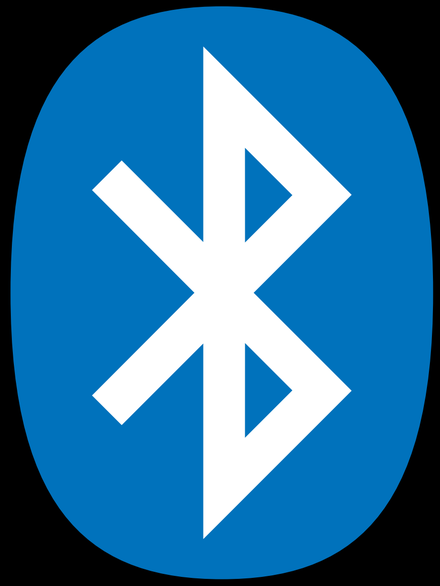
boring facts about

~~fun with~~



# bluetooth

*fun with*



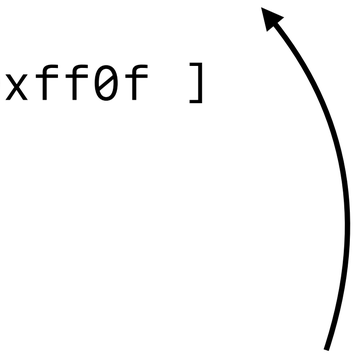
***bluetooth***

# web bluetooth api

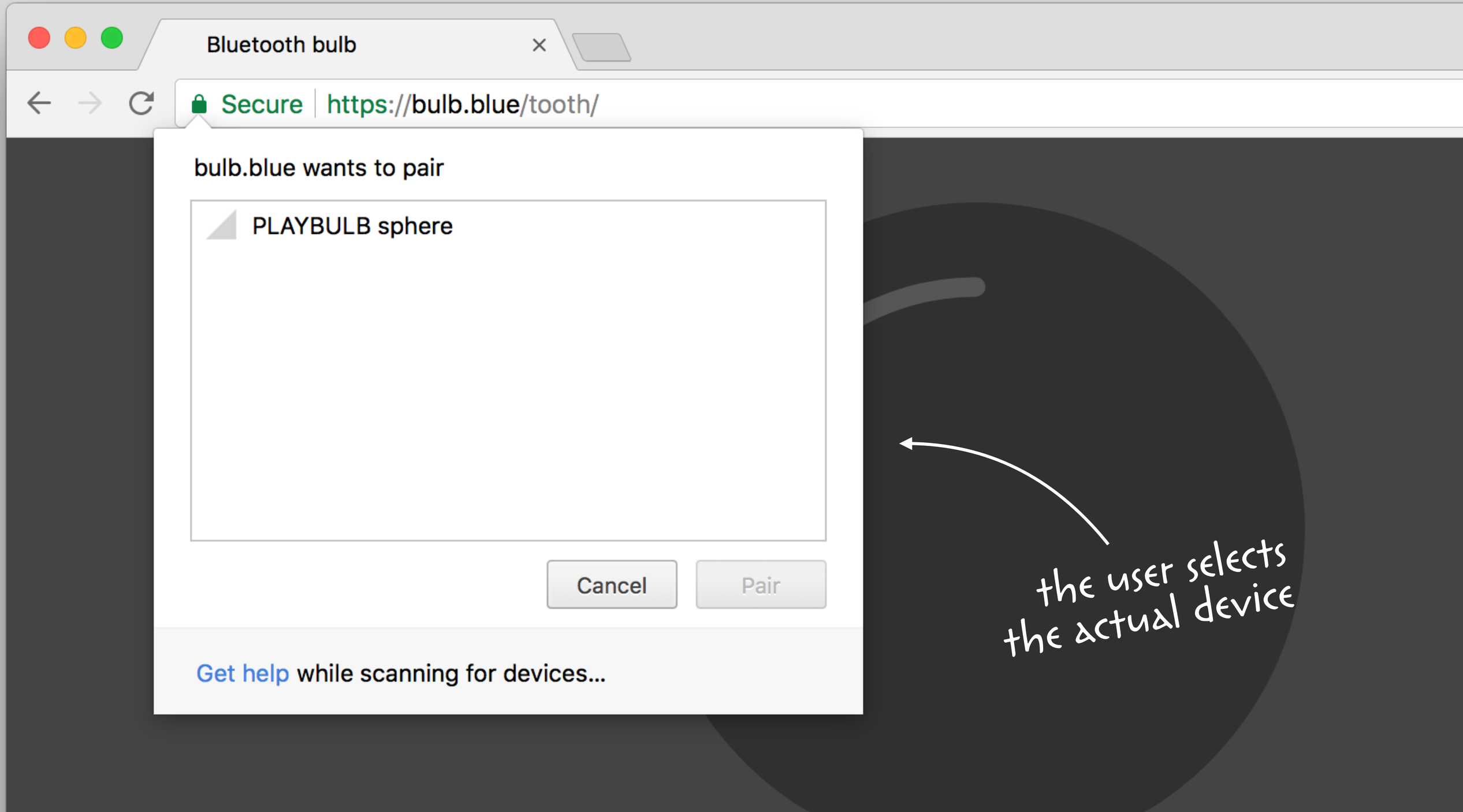
*connecting to a device*

1

```
navigator.bluetooth.requestDevice({
    filters: [
        { namePrefix: 'PLAYBULB' }
    ],
    optionalServices: [ 0xff0f ]
})
```



we tell the browser what kind of device we want

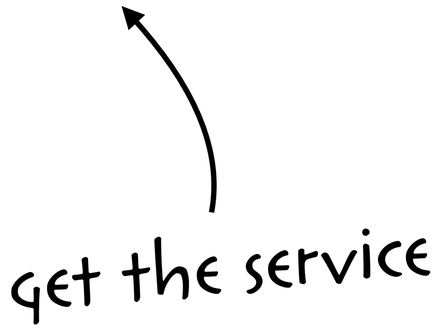

Bluetooth bulb
Secure
https://bulb.blue/tooth/
bulb.blue wants to pair
PLAYBULB sphere
Cancel
Pair
Get help while scanning for devices...
the user selects
the actual device

2

```
navigator.bluetooth.requestDevice({
    filters: [
        { namePrefix: 'PLAYBULB' }
    ],
    optionalServices: [ 0xff0f ]
})

.then(device => device.gatt.connect())
```

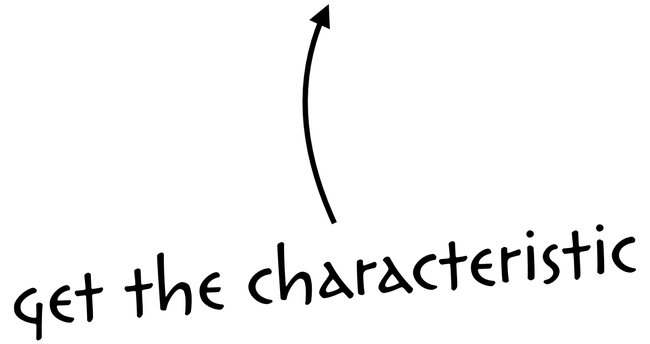
3

```
navigator.bluetooth.requestDevice({
    filters: [
        { namePrefix: 'PLAYBULB' }
    ],
    optionalServices: [ 0xff0f ]
})

.then(device => device.gatt.connect())
.then(server => server.getPrimaryService(0xff0f))
```



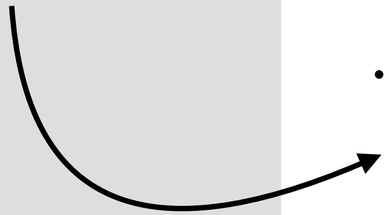
# 4

```
navigator.bluetooth.requestDevice({
    filters: [
        { namePrefix: 'PLAYBULB' }
    ],
    optionalServices: [ 0xff0f ]
})

.then(device => device.gatt.connect())
.then(server => server.getPrimaryService(0xff0f))
.then(service => service.getCharacteristic(0xfffc))
```



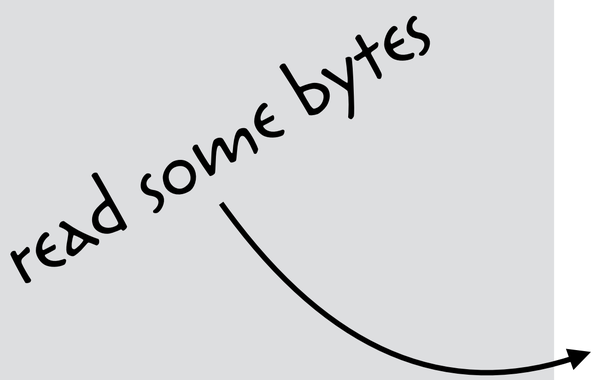
*writing data*

write some bytes



```
navigator.bluetooth.requestDevice({ ... })
.then(device => device.gatt.connect())
.then(server => server.getPrimaryService(0xff0f))
.then(service => service.getCharacteristic(0xfffc))

.then(c => {
    return c.writeValue(
        new Uint8Array([ 0x00, r, g, b  ])
    );
})
```

# *reading data*




```
navigator.bluetooth.requestDevice({ ... })
.then(device => device.gatt.connect())
.then(server => server.getPrimaryService(0xff0f))
.then(service => service.getCharacteristic(0xfffc))

.then(c => c.readValue())
.then(value => {
    let r = value.getUint8(1);
    let g = value.getUint8(2);
    let b = value.getUint8(3);
})
```

*get notified of changes*

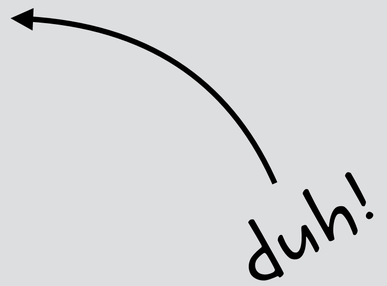
add event listener

```
navigator.bluetooth.requestDevice({ ... })
.then(device => device.gatt.connect())
.then(server => server.getPrimaryService(0xff0f))
.then(service => service.getCharacteristic(0xfffc))

.then(c => {
    c.addEventListener('characteristicvaluechanged', e => {
        let r = e.target.value.getUint8(1);
        let g = e.target.value.getUint8(2);
        let b = e.target.value.getUint8(3);
    });

    c.startNotifications();
})
```

don't forget to start listening

# *things you need to know:*

- the webbluetooth api
- promises
- typed arrays



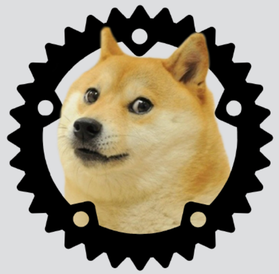
# *browser support*

Chrome

Opera

Samsung

(behind flag)



Servo

(soon)

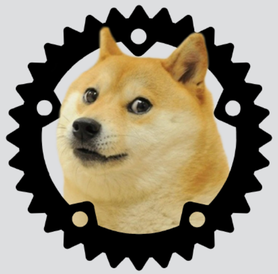
# *browser support*

kinda works

Chrome

Opera

Samsung
(behind flag)



Servo
(soon)

WebBLE
for iOS

*and...* 

```
npm install node-web-bluetooth
```

# *and... the puck.js*

*custom characteristics* *wtf!*

writing a value:

```
function(r, g, b) {
    return new Uint8Array([ 0x00, r, g, b  ]);
}
```

reading a value:

```
function(buffer) {
    return {
        r: buffer.getUint8(1),
        g: buffer.getUint8(2),
        b: buffer.getUint8(3)
    }
}
```

writing to and reading
from the same characteristic

## writing a value:

```
function(r, g, b) {
    return new Uint8Array([
        0x01, g, 0x01, 0x00, 0x01,
        b, 0x01, r, 0x01, 0x00
    ]);
}
```


reading the current color is not possible

## writing a value:

```
function(r, g, b) {
    var buffer = new Uint8Array([
        0xaa, 0x0a, 0xfc, 0x3a, 0x86, 0x01, 0x0d,
        0x06, 0x01, r, g, b, 0x00, 0x00,
        (Math.random() * 1000) & 0xff, 0x55, 0x0d
    ]);

    for (var i = 1; i < buffer.length - 2; i++) {
        buffer[15] += buffer[i];
    }

    return buffer;
}
```

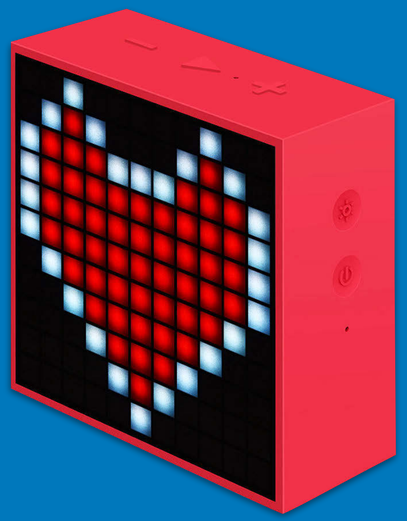

reading the current color is not possible

## writing a value:

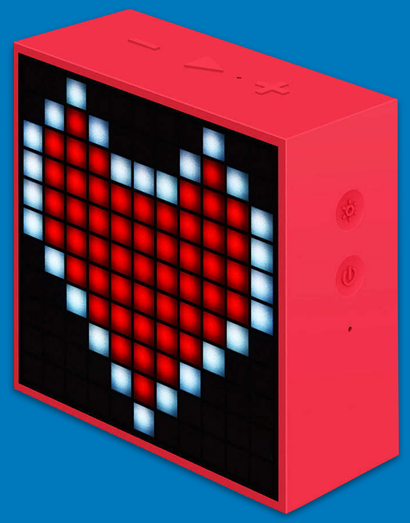
```
function(r, g, b, position) {
    let buffer = new Uint8Array([
        0x07, 0x02, position + 1, r, g, b
    ]);

    return buffer;
}
```

writing a value:

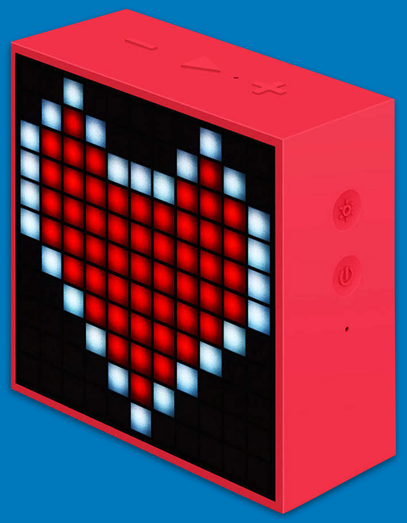
```
function(r, g, b, position) {
    let buffer = new Uint8Array([
        0x58, r, g, b, 0x01, position
    ]);

    ...
```



writing a value:



```
function(r, g, b, position) {
    let buffer = new Uint8Array([
        0x58, r, g, b, 0x01, position
    ]);

    let payload = new Uint8Array(buffer.length + 4);
    payload[0] = payload.length - 2;
    payload[1] = payload.length - 2 >>> 8;
    payload.set(buffer, 2);

    let checksum = payload.reduce((a, b) => a + b, 0);
    payload[payload.length - 2] = checksum;
    payload[payload.length - 1] = checksum >>> 8;

    let extra = payload.filter(value => {
```



```
                message[m + 1] = 0x05;
                m += 2;
            }
            else if (payload[i] === 0x03) {
                message[m] = 0x03;
                message[m + 1] = 0x06;
                m += 2;
            }
            else {
                message[m] = payload[i];
                m++;
            }
        }

        message[0] = 0x01;
        message[message.length - 1] = 0x02;

        return message;
    }
```

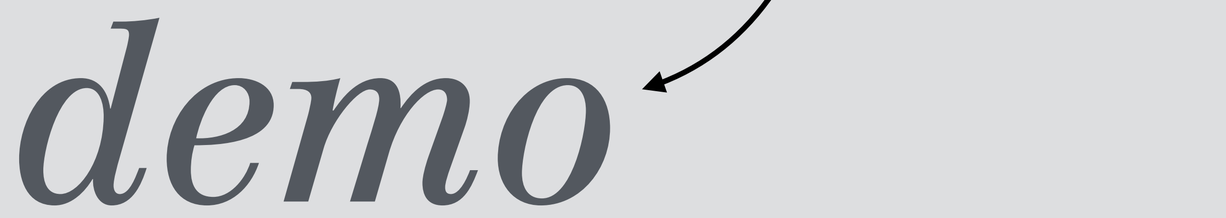

# *adafruit bluetooth sniffer*

*decompiling*

# *the apk*

don't tell anyone!

finally the fun part



*warning*

experimental technology

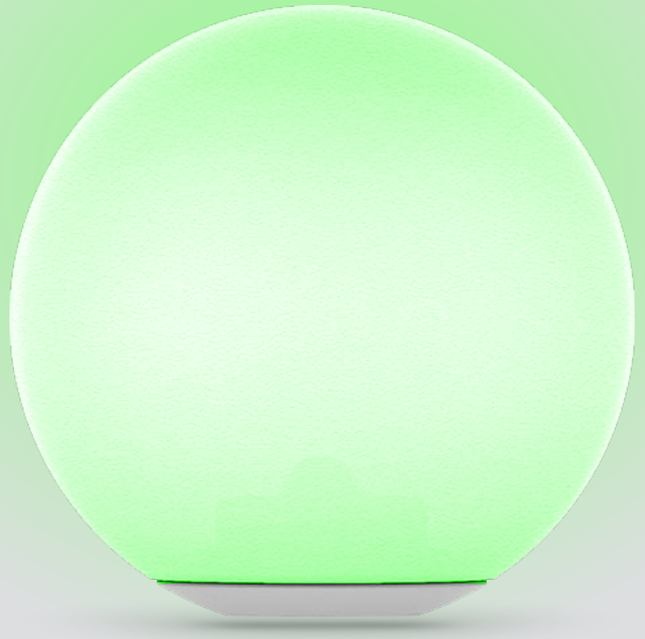
setting low expectations

*warning*

wifi interference

lowering them even further

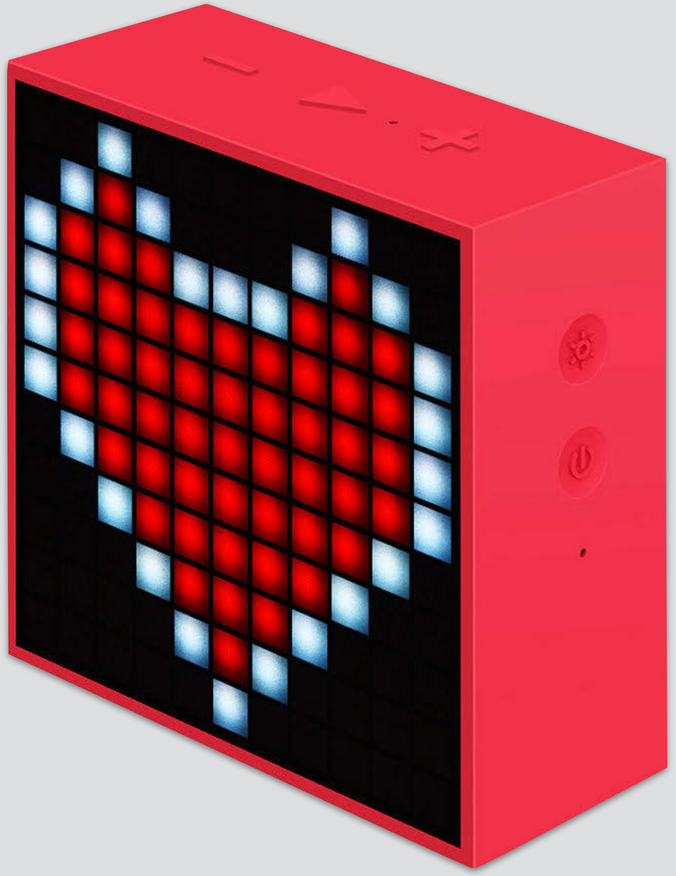
*change the colour*
*of a lightbulb*



https://bluetooth.rocks/lightbulb

https://github.com/NielsLeenheer/BluetoothBulb

*draw pixel art on*
*a led matrix display*



https://bluetooth.rocks/pixel

*control a lego racer*
*using a gamepad*

*use css animations to*
*define a path*


https://bluetooth.rocks/racer

https://github.com/NielsLeenheer/BluetoothRacer

*control a drone*
*from your browser*


https://bluetooth.rocks/drone

https://github.com/poshaughnessy/web-bluetooth-parrot-drone

# *print on a receipt printer*

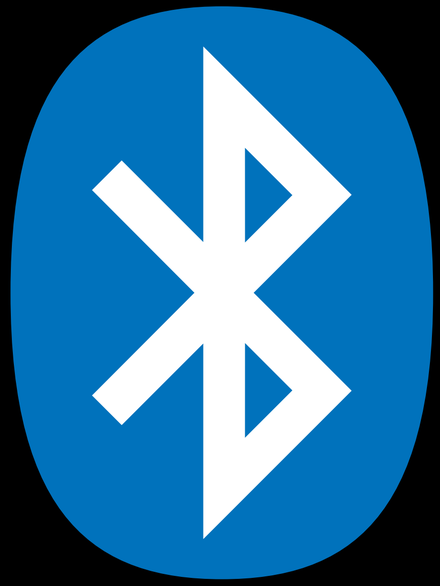

https://bluetooth.rocks/printer

*find out your*
*current heartbeat*


https://bluetooth.rocks/pulse

https://github.com/NielsLeenheer/BluetoothPulse

*fun with*



***bluetooth!***

# *questions?*