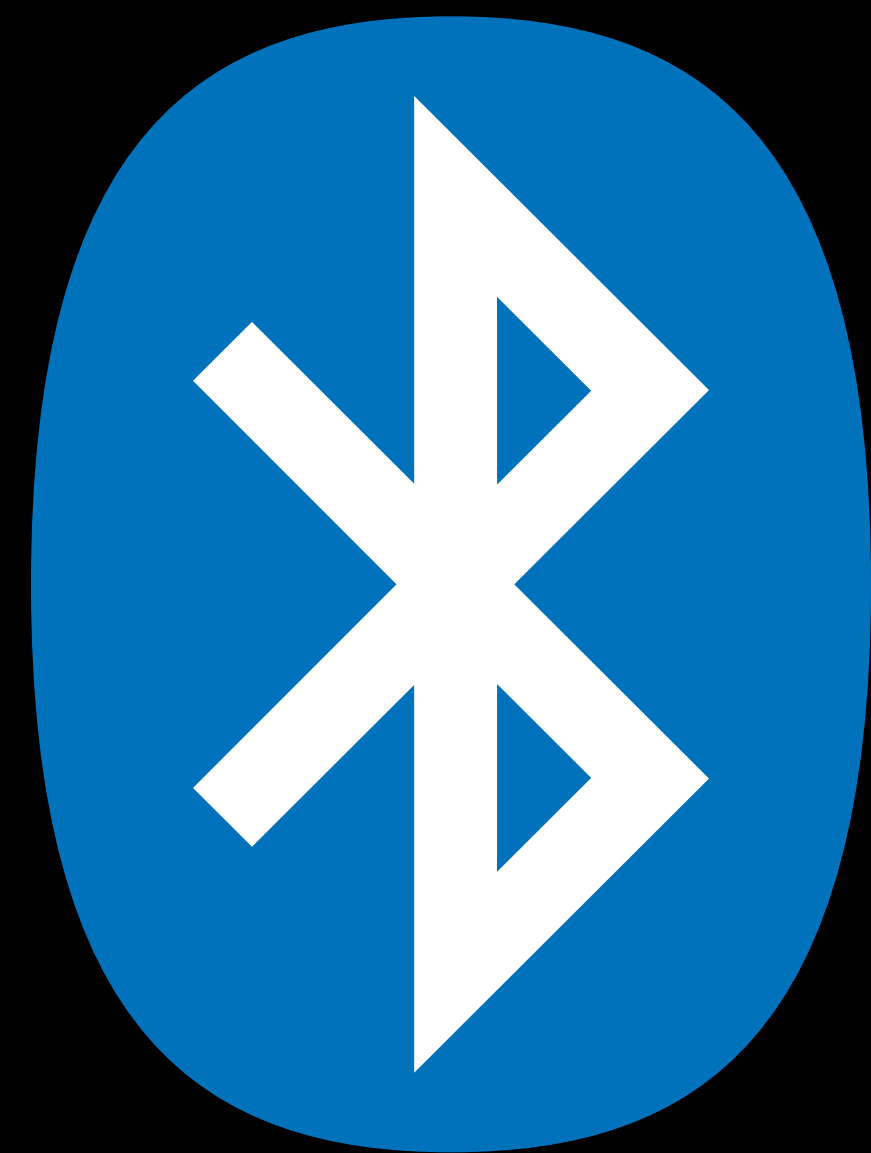


fun with



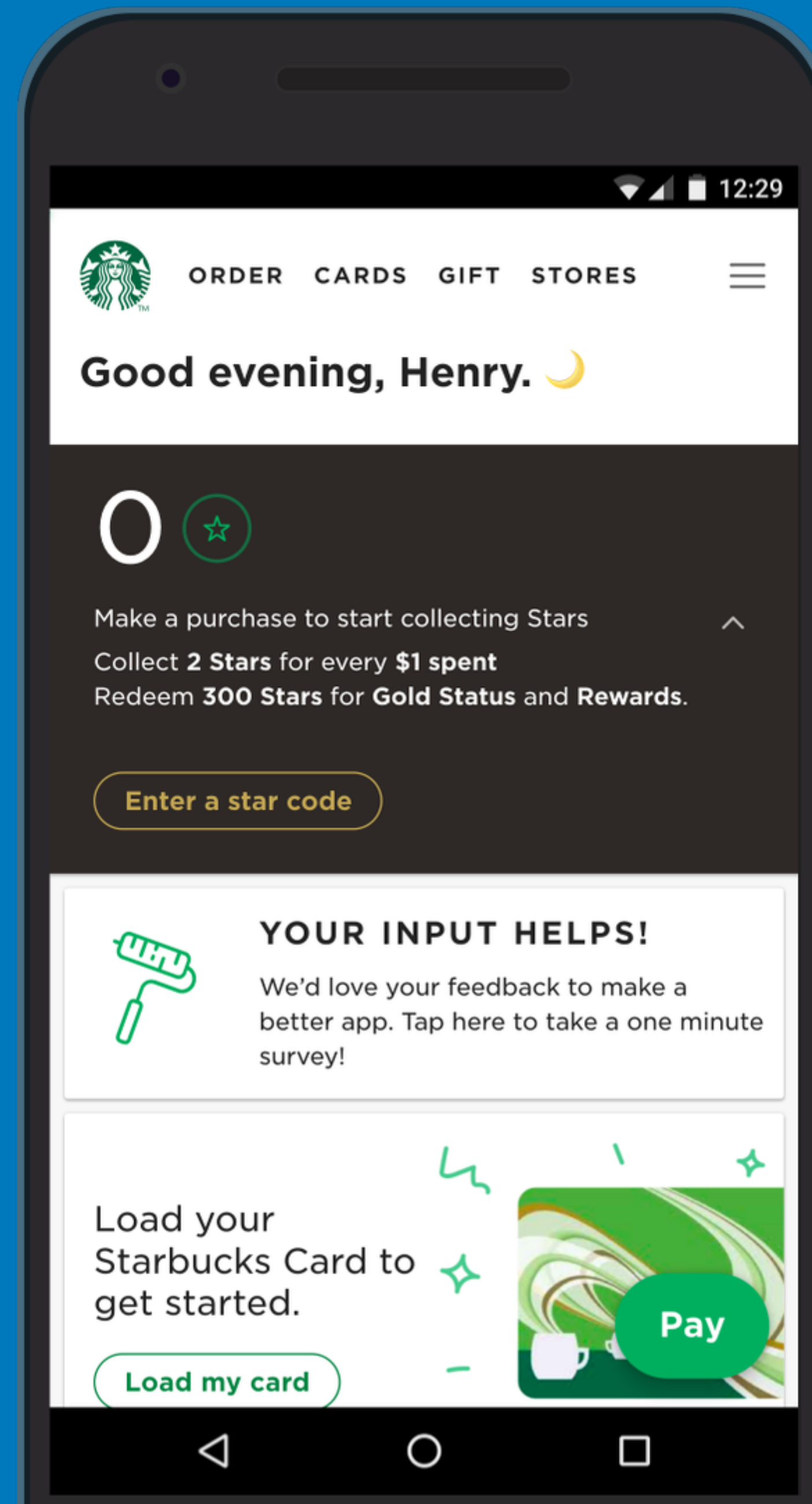
bluetooth

why?

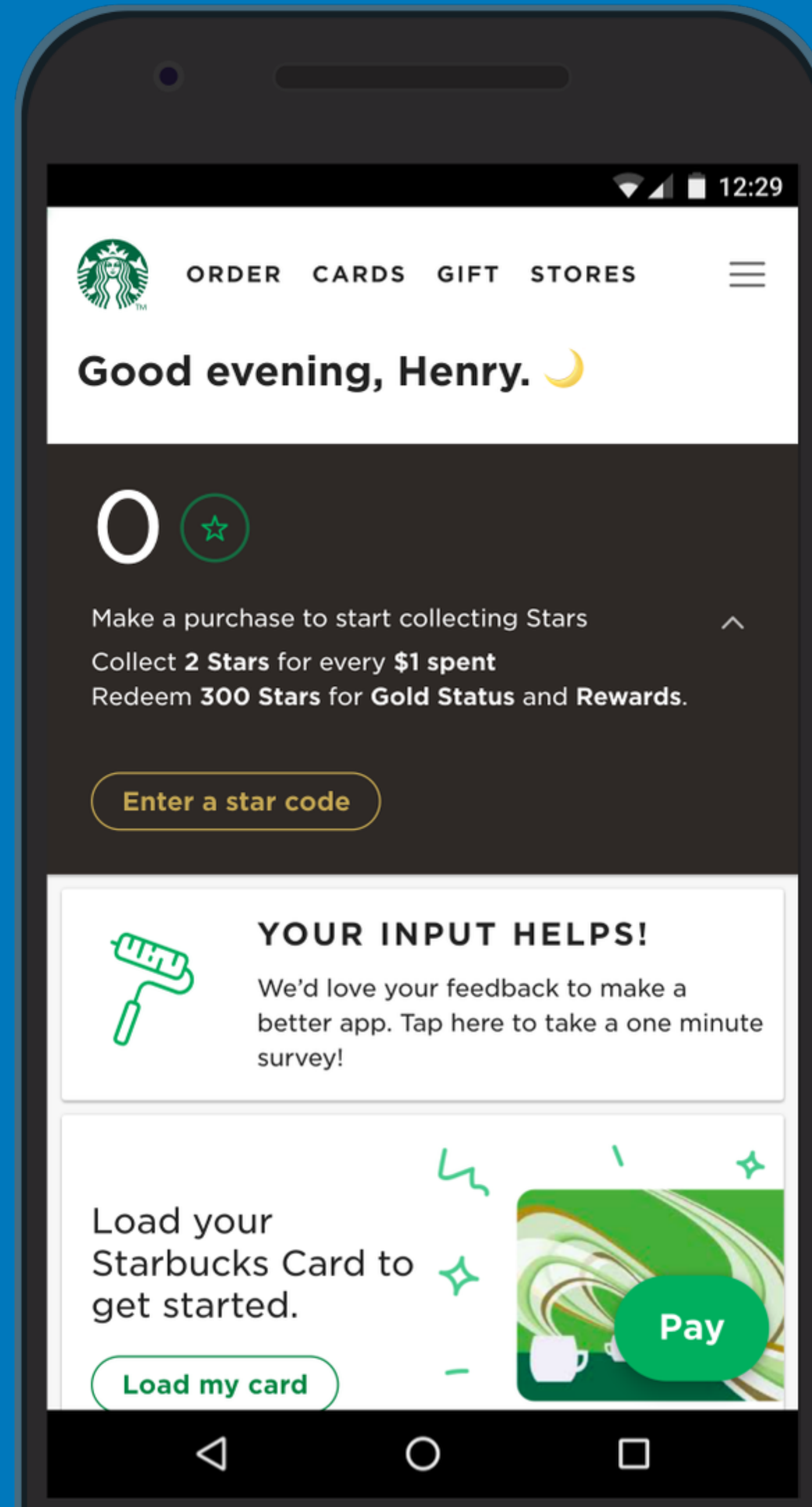


*progressive
web apps*

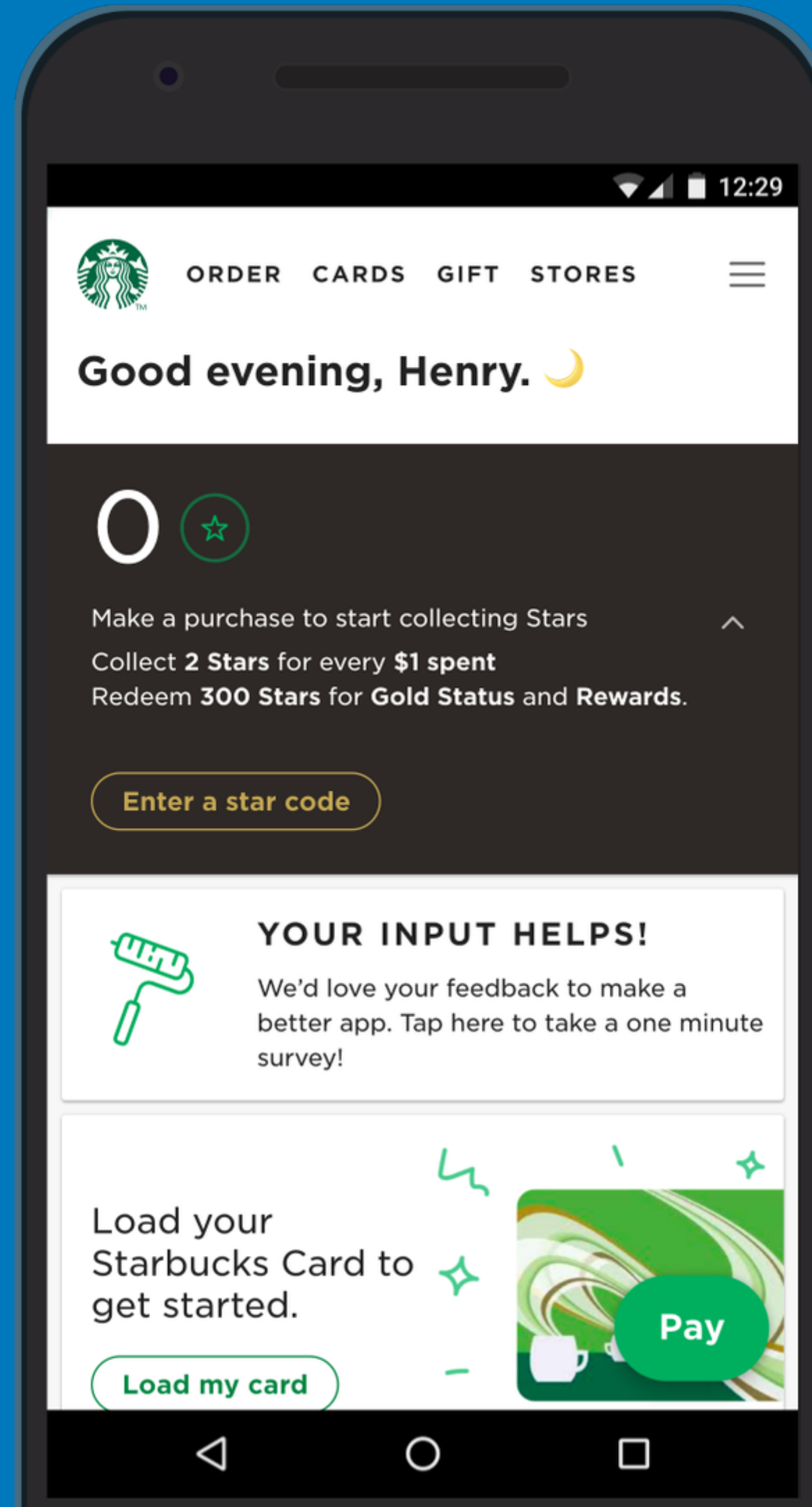
*pwa's
are great!*



but...



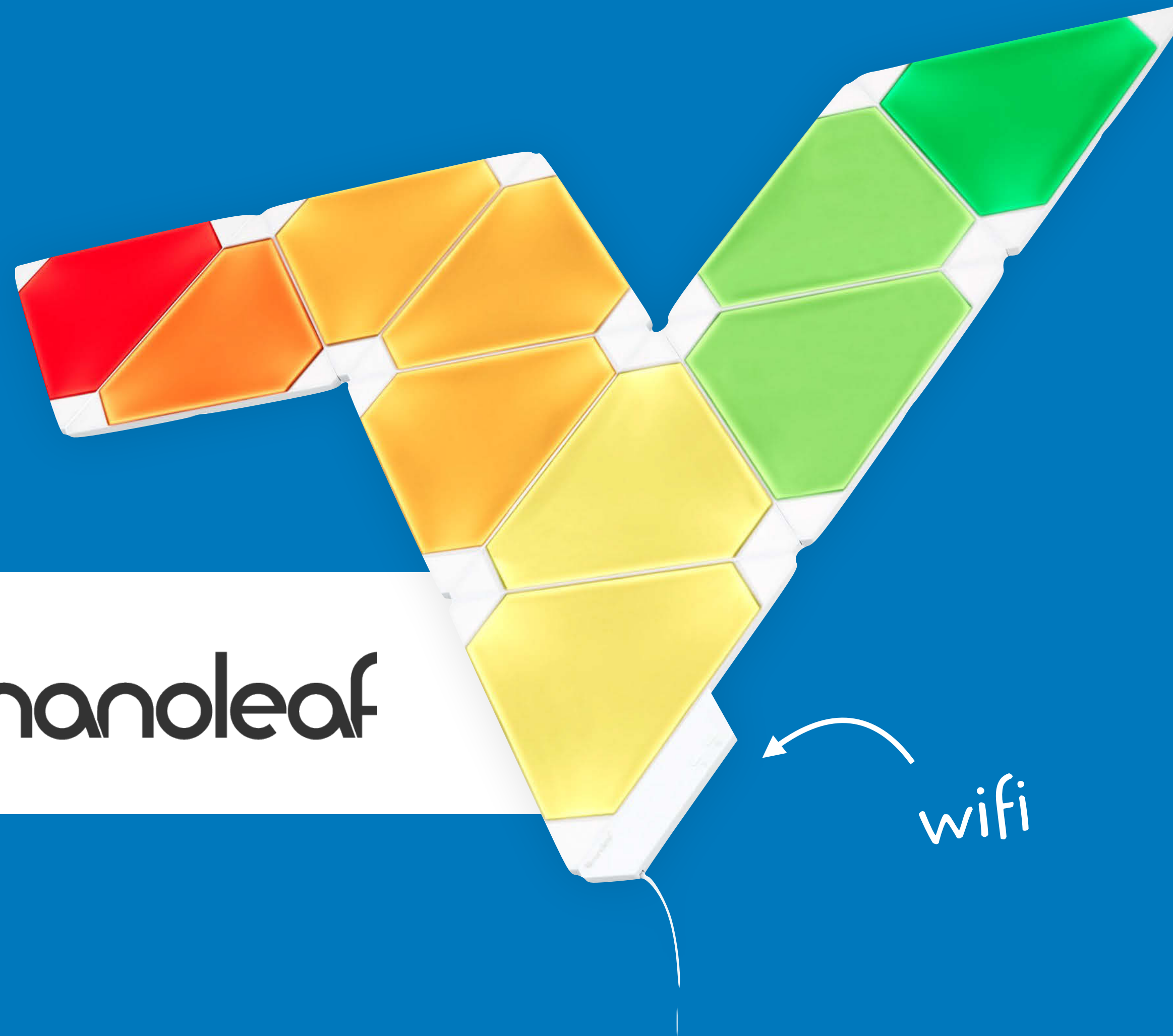
but...



ethernet
& wifi



SONOS



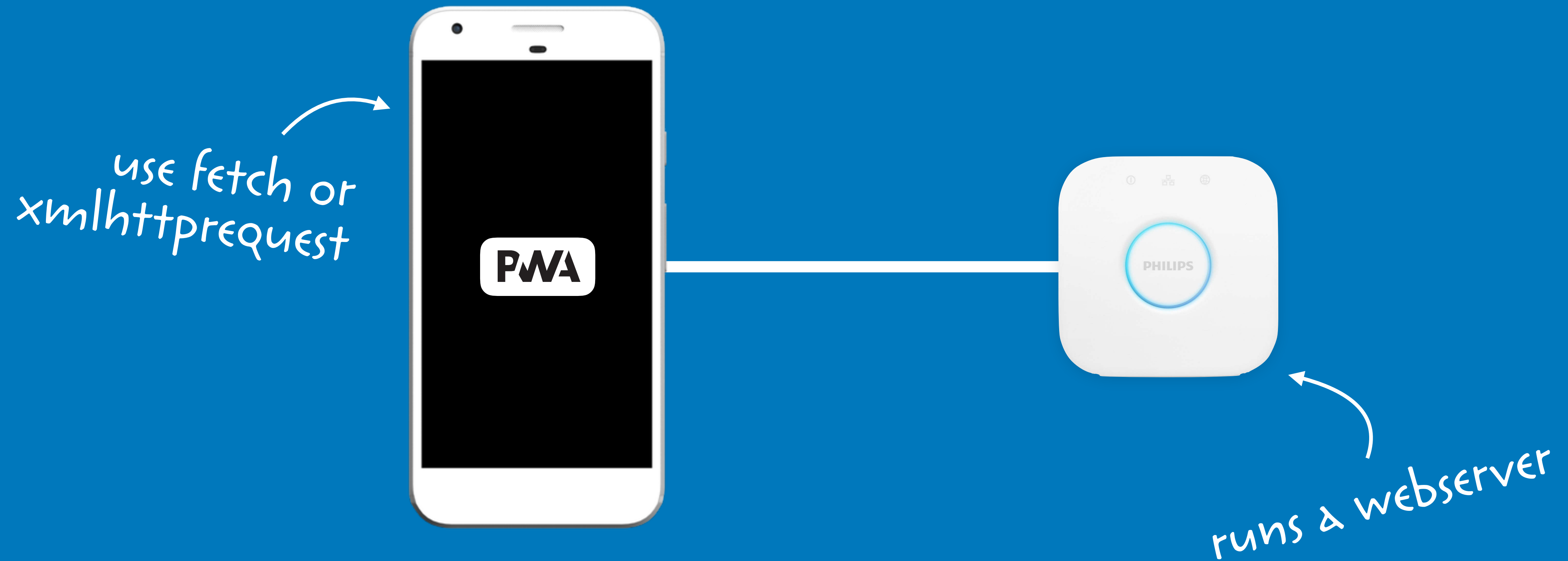
 nanoleaf

zigbee?



hue





1

`https://www.meethue.com/api/nupnp`



```
[{
  "id": "001788ffffe100491",
  "internalipaddress": "192.168.2.23",
  "macaddress": "00:17:88:10:04:91",
  "name": "Philips Hue"
},
{
  "id": "001788ffffe09a168",
  "internalipaddress": "192.168.88.252"
},
]
```

2

`http://192.168.2.23/description.xml`



```
<root xmlns="urn:schemas-upnp-org:device-1-0">
  <specVersion>
    <major>1</major>
    <minor>0</minor>
  </specVersion>
  <URLBase>http://192.168.2.23:80/</URLBase>
  <device>
    <deviceType>urn:schemas-upnp-org:device:Basic:1</deviceType>
    <friendlyName>Philips hue (192.168.2.23)</friendlyName>
    <manufacturer>Royal Philips Electronics</manufacturer>
```

3

```
http://192.168.2.23/api
```

```
{"devicetype": "hue_pwa#pixel_xl"}
```



User presses physical button on the hub



click

```
[{"success": {"username": "....."}}
```

save to local storage

3

```
http://192.168.2.23/api
```

```
{"devicetype": "hue_pwa#pixel_xl"}
```



User presses physical button on the hub



click

```
[{"success": {"username": "....."}}
```

save to local storage

4

`http://192.168.2.23/api/...../lights`



```
{
  "1": {
    "state": {
      "on": true,
      "bri": 144,
      "hue": 13088,
      "sat": 212,
      "xy": [0.5128, 0.4147],
      "ct": 467,
      "alert": "none",
```

5

`http://192.168.2.23/api/...../lights/1/state`

```
{  
  "hue": 50000,  
  "on": true,  
  "bri": 200  
}
```



```
[  
  {"success": {"/lights/1/state/bri": 200}},  
  {"success": {"/lights/1/state/on": true}},  
  {"success": {"/lights/1/state/hue": 50000}}  
]
```

click

click

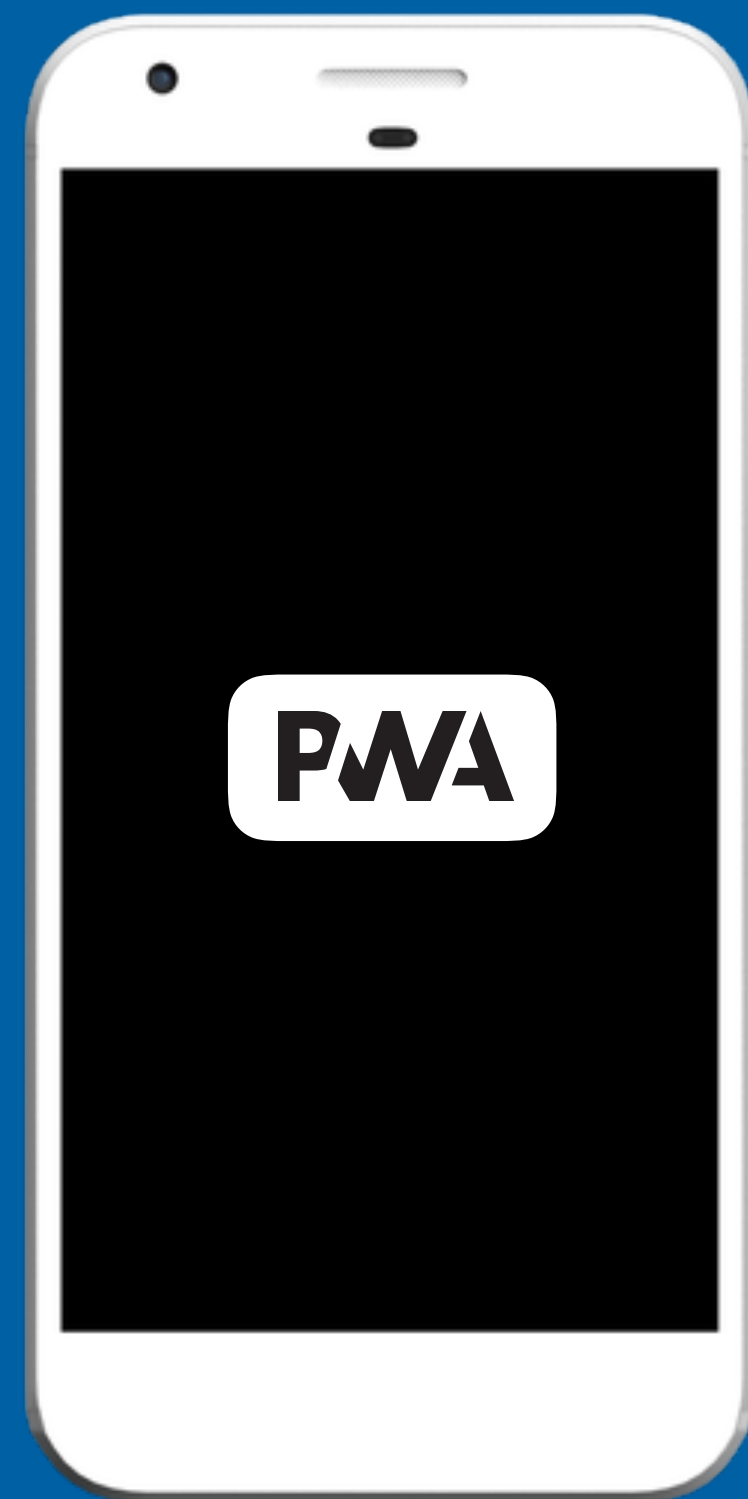


reload

click

this does !
not work !





https only



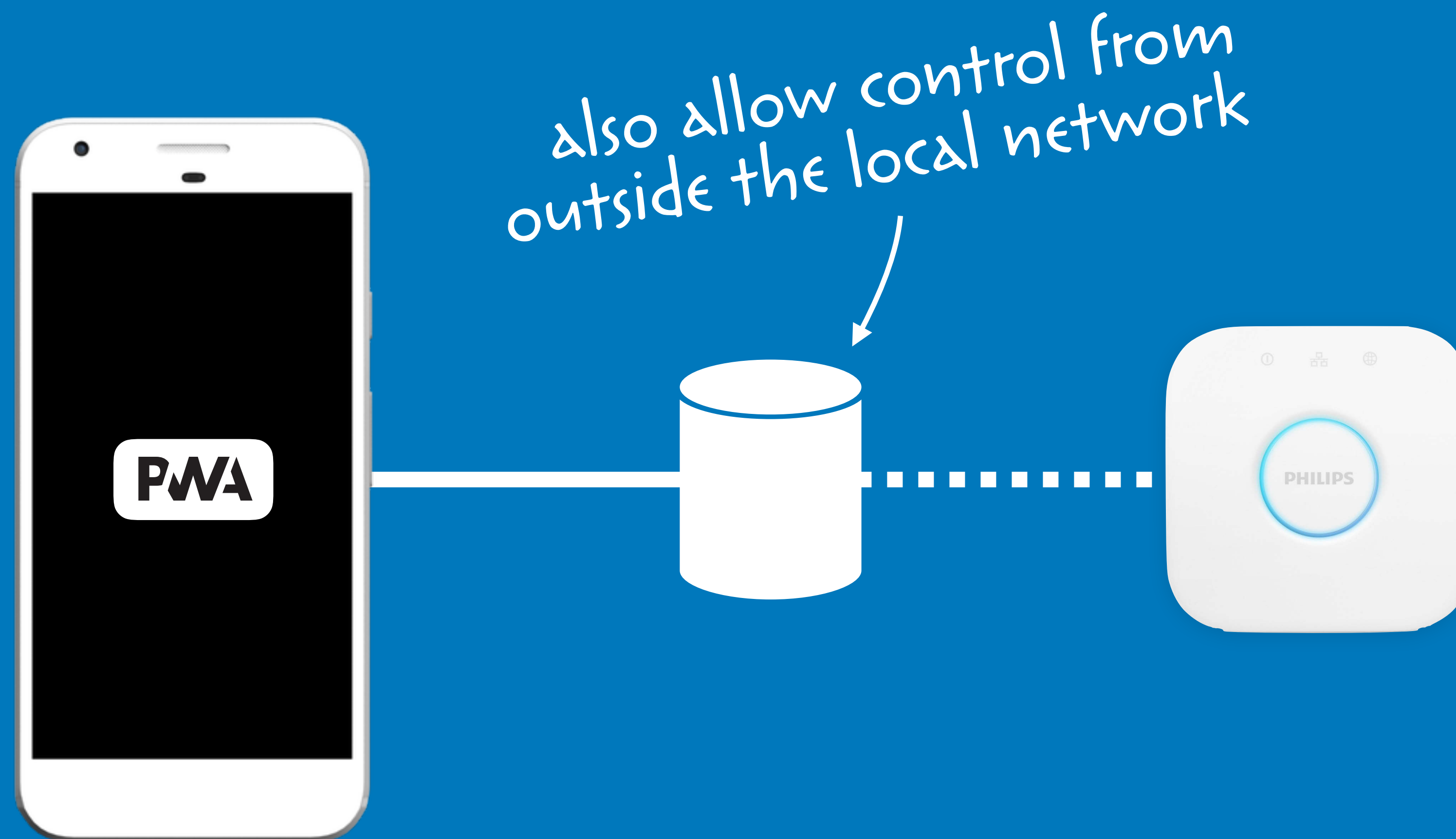
*no certificate
for you*

http only

network

= use remote api's

≠ access to local devices



hue remote api

“

We are currently preparing for release of this remote API which will initially be available to a limited number of partners.

— Philips

two years ago

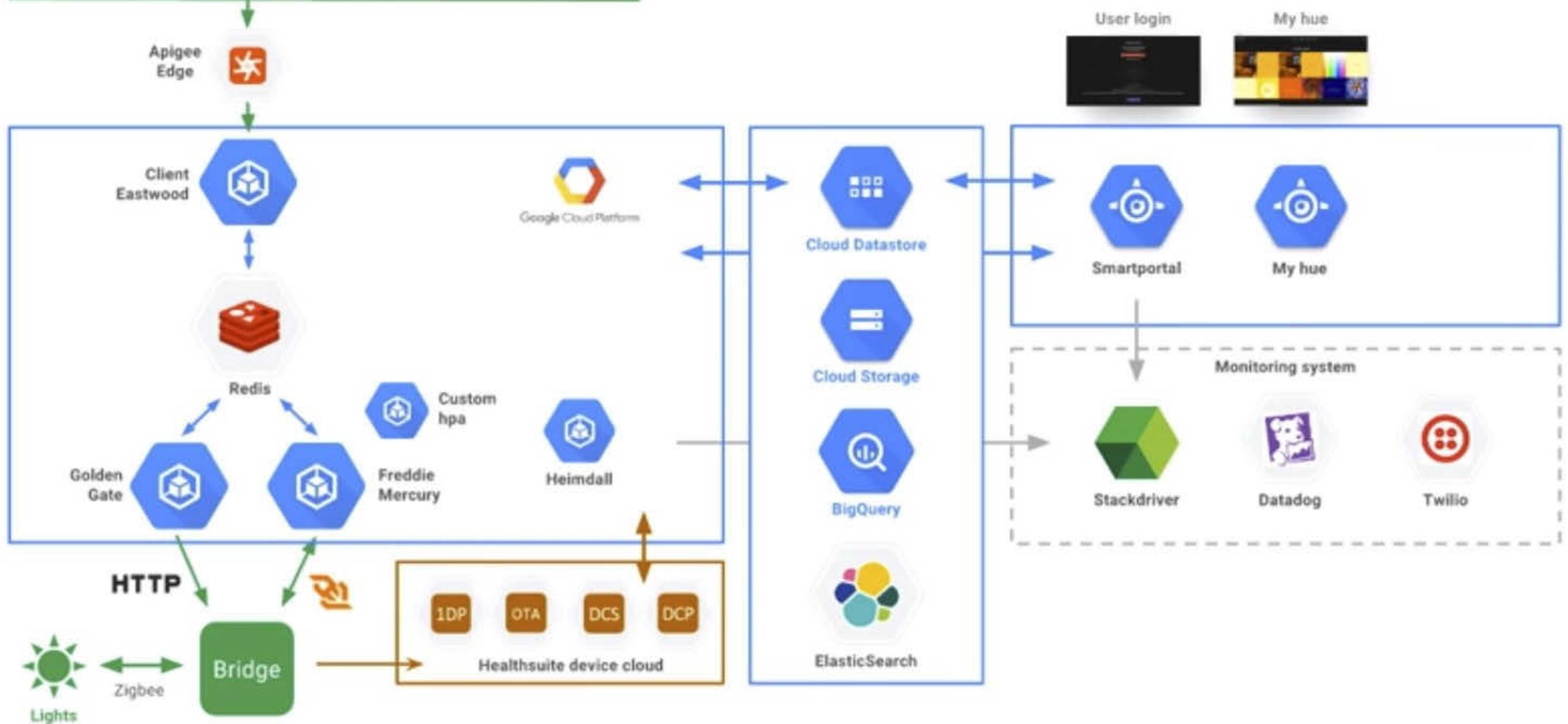
“

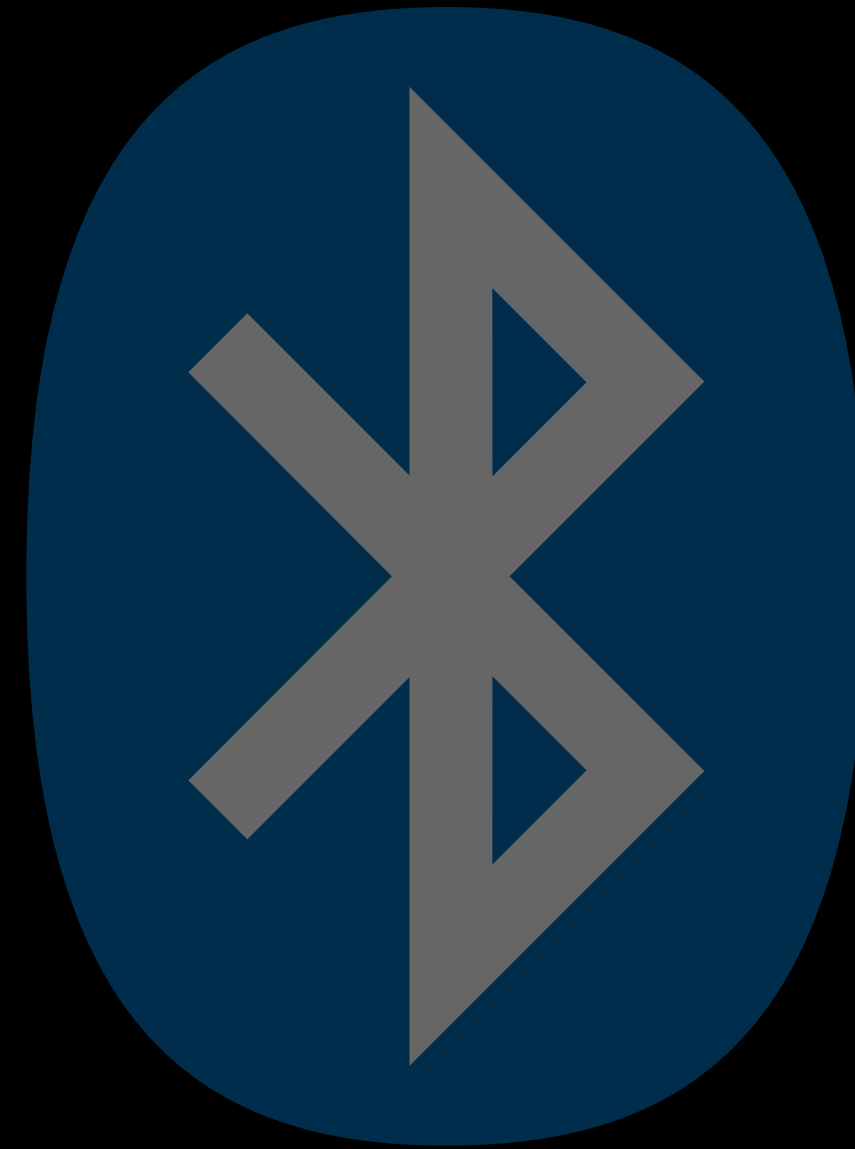
We are currently preparing for release of this remote API which will initially be available to a very limited number of partners.

— Philips

just nest,
google assistant
and ifttt

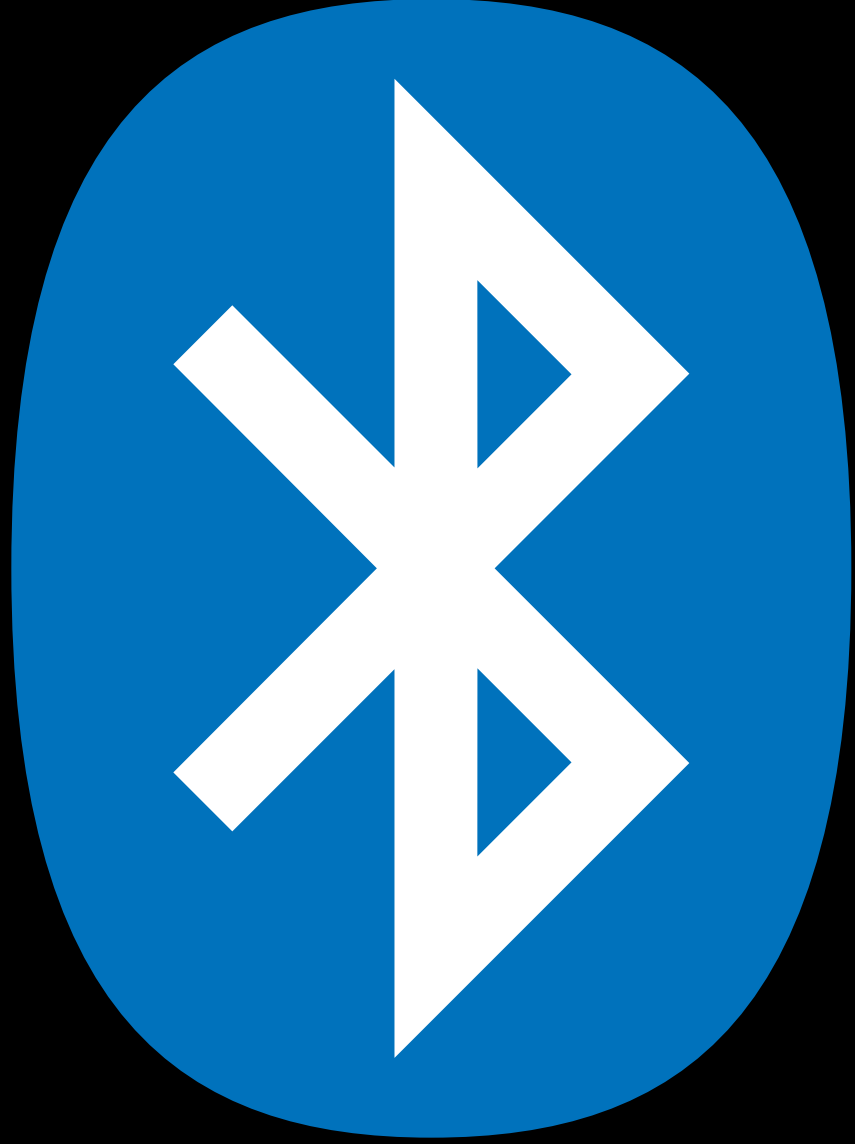
3rd party API clients
(Nest, Google Assistant, IFTTT, 3rd party apps, ...)

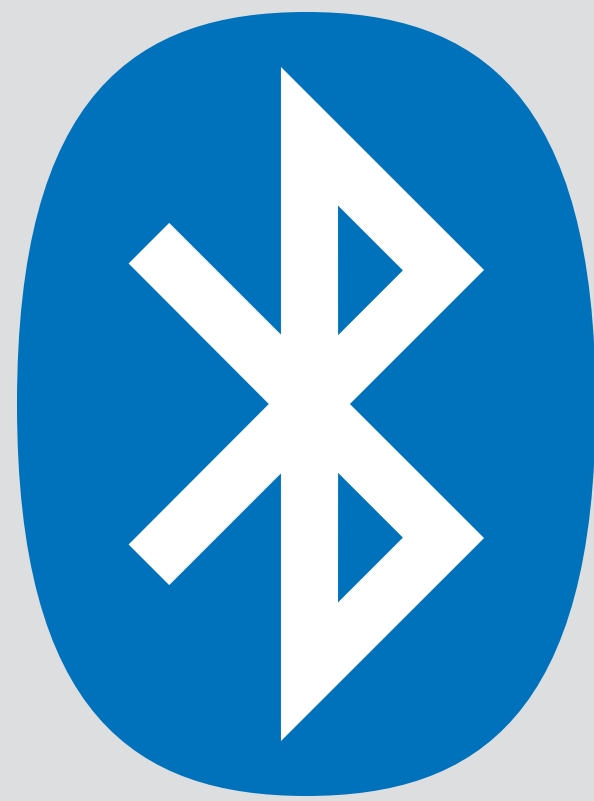




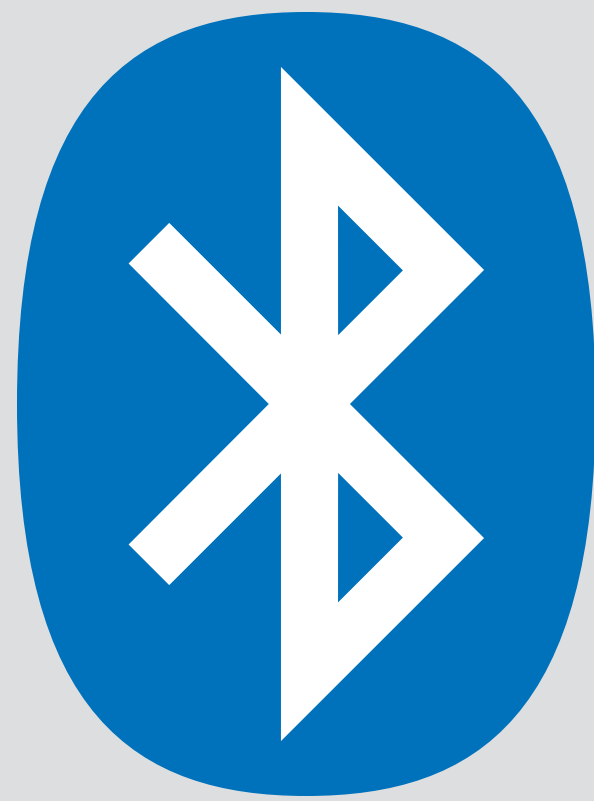
problems
~~fun~~ with

~~bluetooth~~
local networks

fun with  *bluetooth*

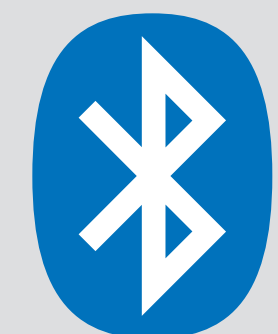


bluetooth



bluetooth

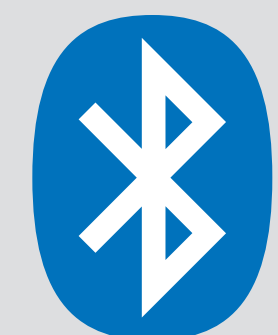
sucks



classic bluetooth

*the reason everybody
hates bluetooth*

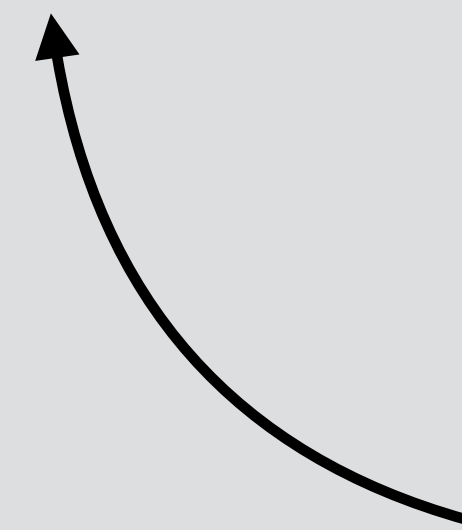
vs.



bluetooth low energy

control drones and other cool shit

bluetooth low energy



also known as

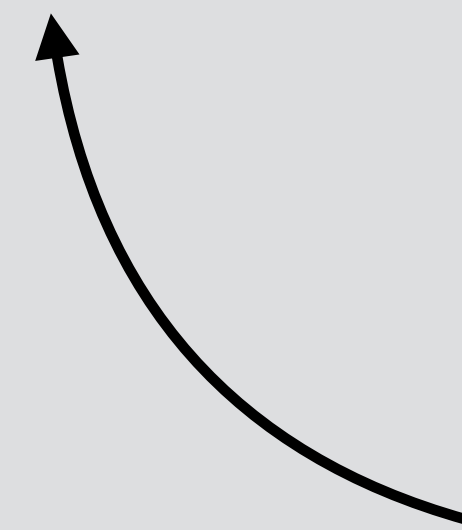
Bluetooth Smart

Bluetooth LE

BLE

Bluetooth 4

bluetooth low energy



also known as

Bluetooth Smart

Bluetooth LE

BLE

Bluetooth 4 and 5

10 million

bluetooth devices
shipping every day

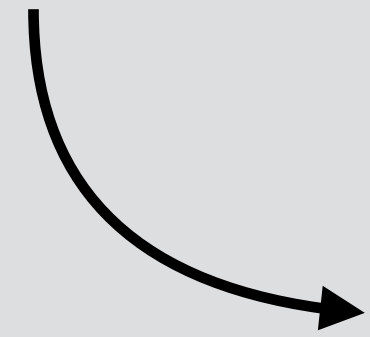


mobile phone

computer



glucose monitor

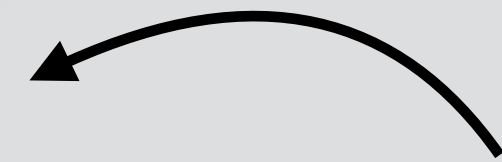
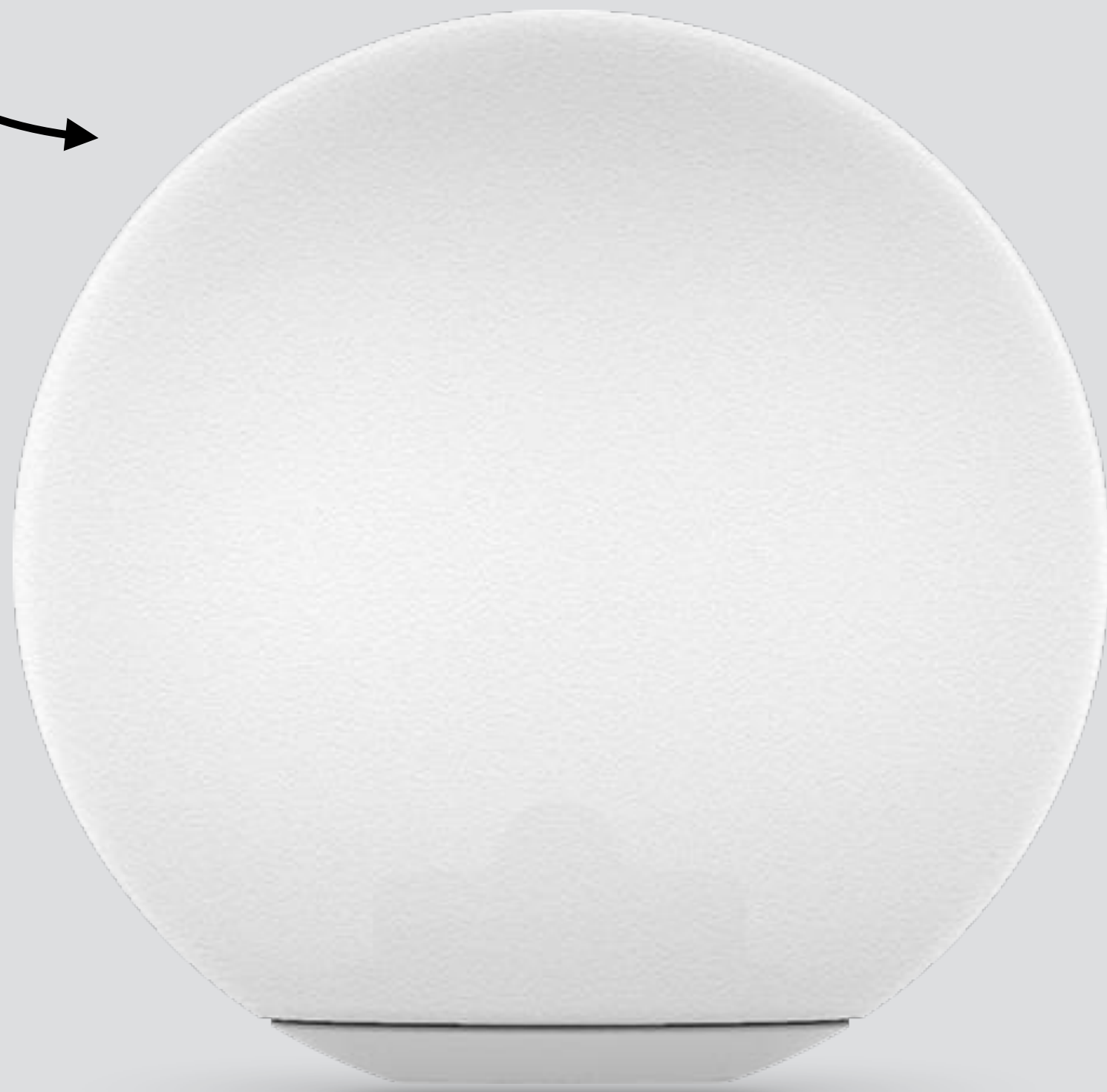
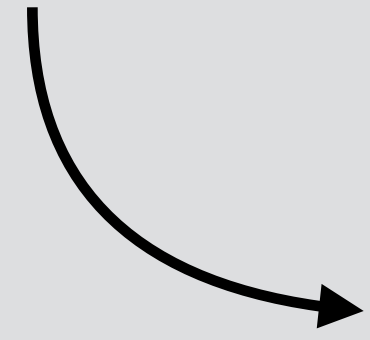


somebody's hand





playbulb sphere



playbulb

spherio bb-8



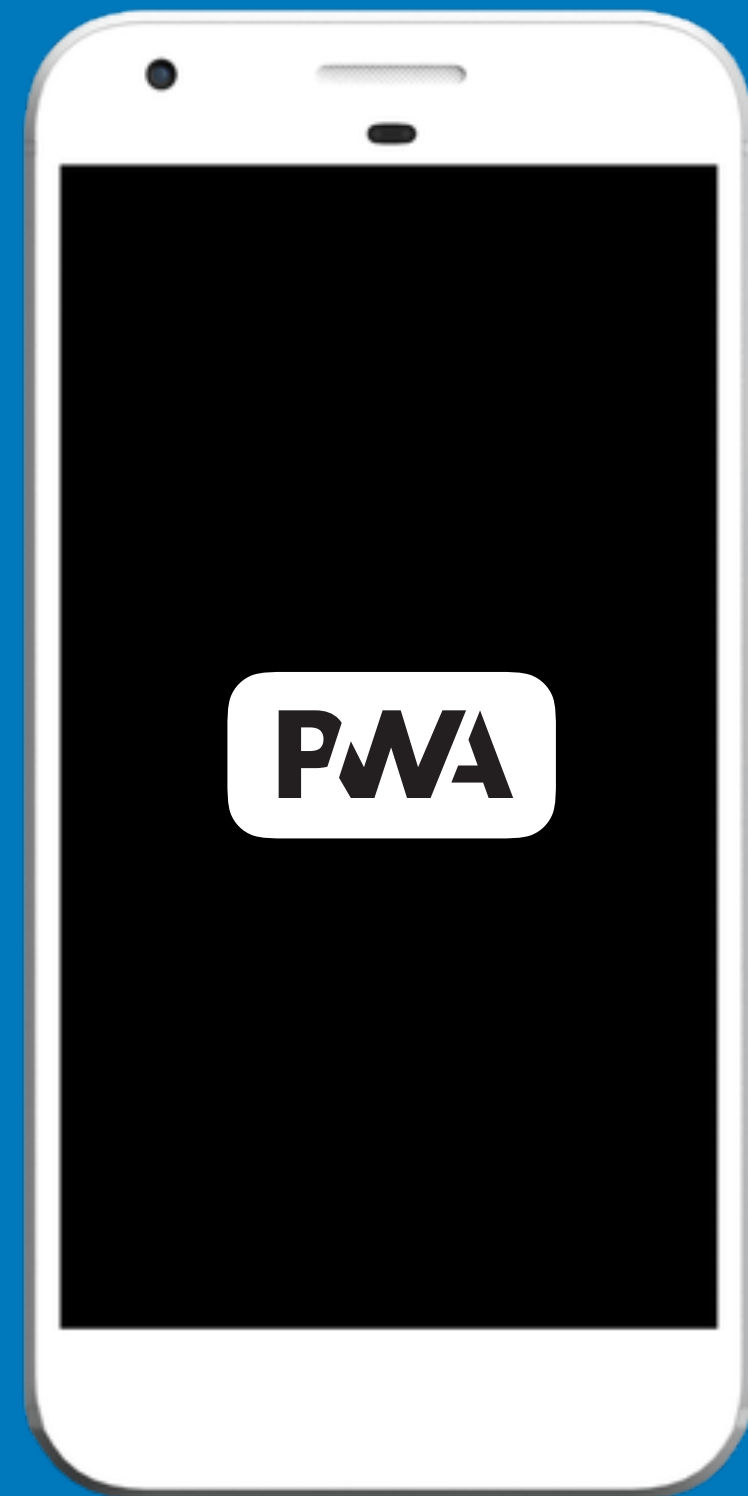


parrot mini drone



fidget spinner

the boring theoretical stuff



central



peripheral



central

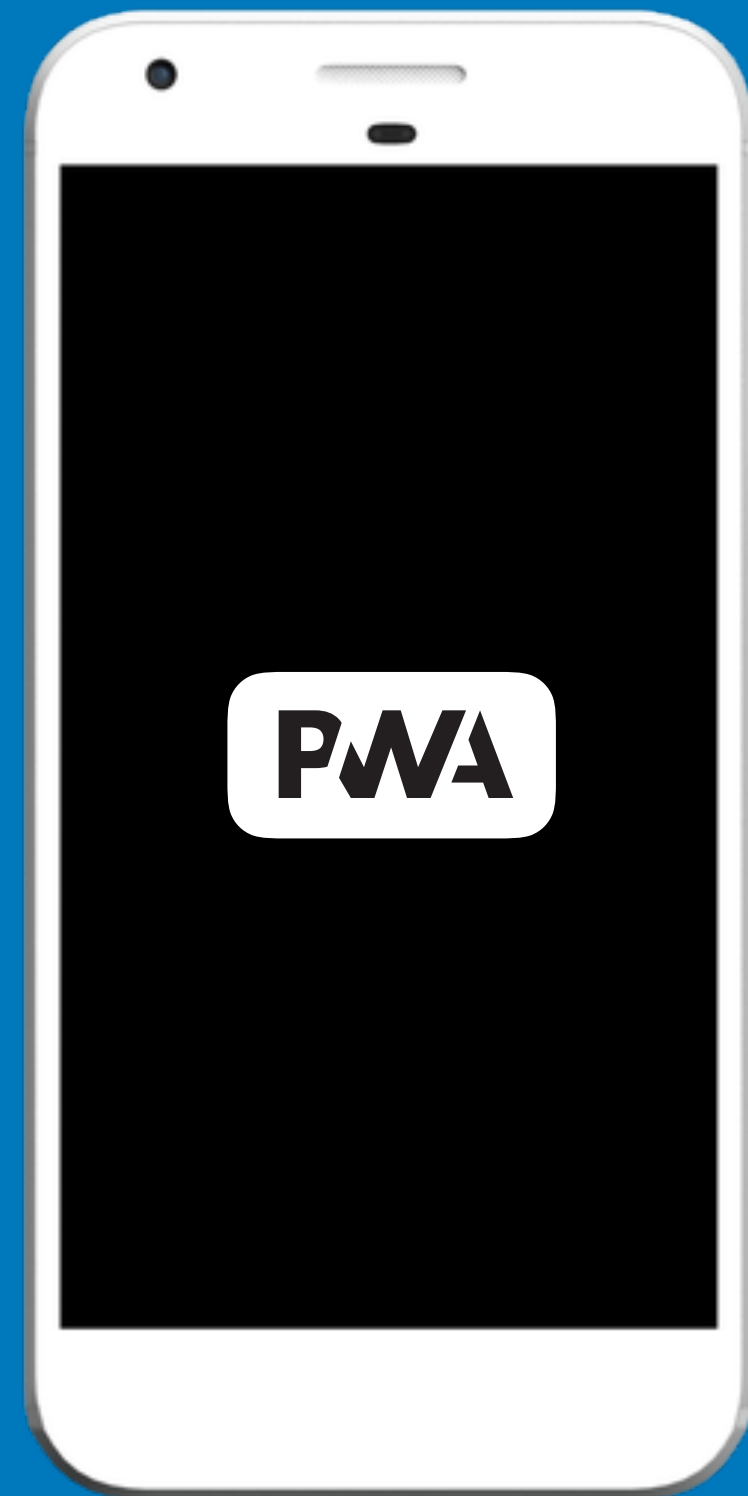
generic attribute profile

generic_attribute_profile ?

generic attribute profile



gatt, because gap was already taken



~~central~~
client



~~peripheral~~
server



device information



light

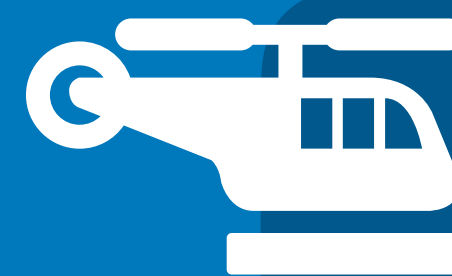
multiple services per device



device information



battery



flight control



device information



battery



steering control



device information



battery



heart rate





device information



battery



heart rate



device information



battery



heart rate



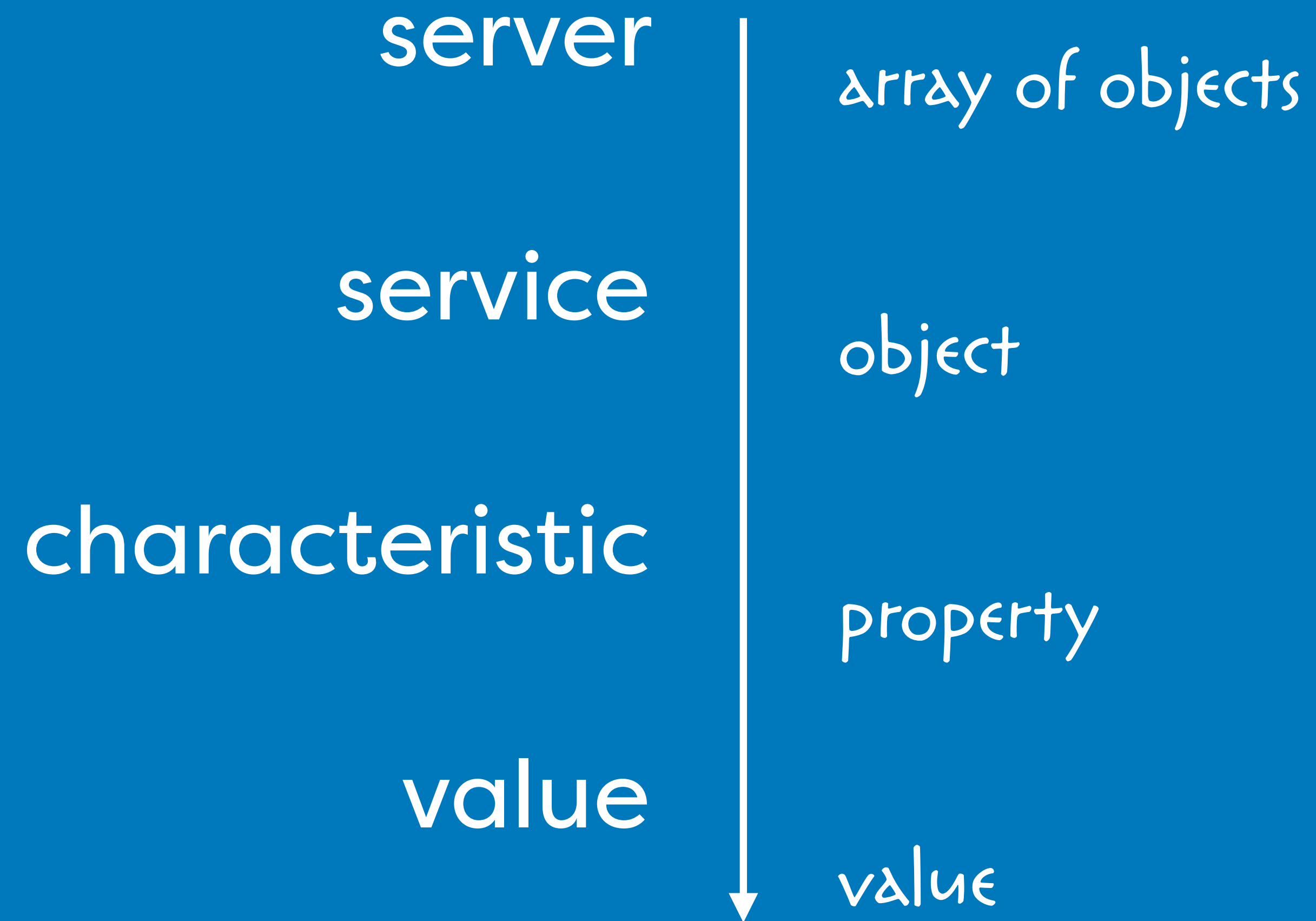


device information

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

*multiple characteristics
per service*





services and characteristics
are identified by uuid's



16 bit or 128 bit



device information



16 bit 0x180A

128 bit 0000180A-0000-1000-8000-00805F9B34FB



battery



16 bit 0x180F

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



light



steering control



flight control

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
...



i

0x180A

0x2A29

0x2A24

0x2A25

0x2A27

0x2A26

0x2A28

...



bad for readability,
good for saving bandwidth

each characteristic supports
one or more of these



read

write

write without response

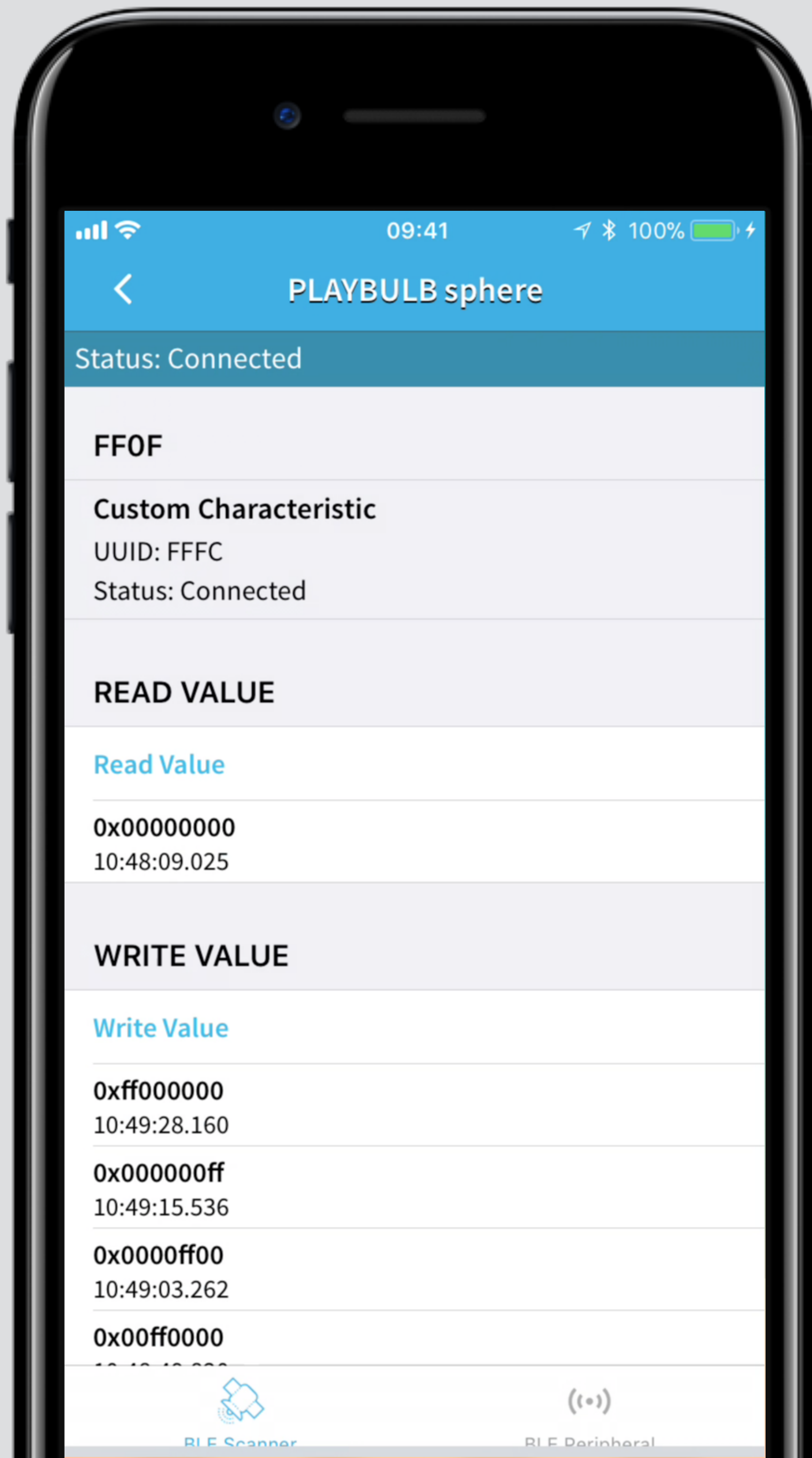
notify

every value is an array of bytes

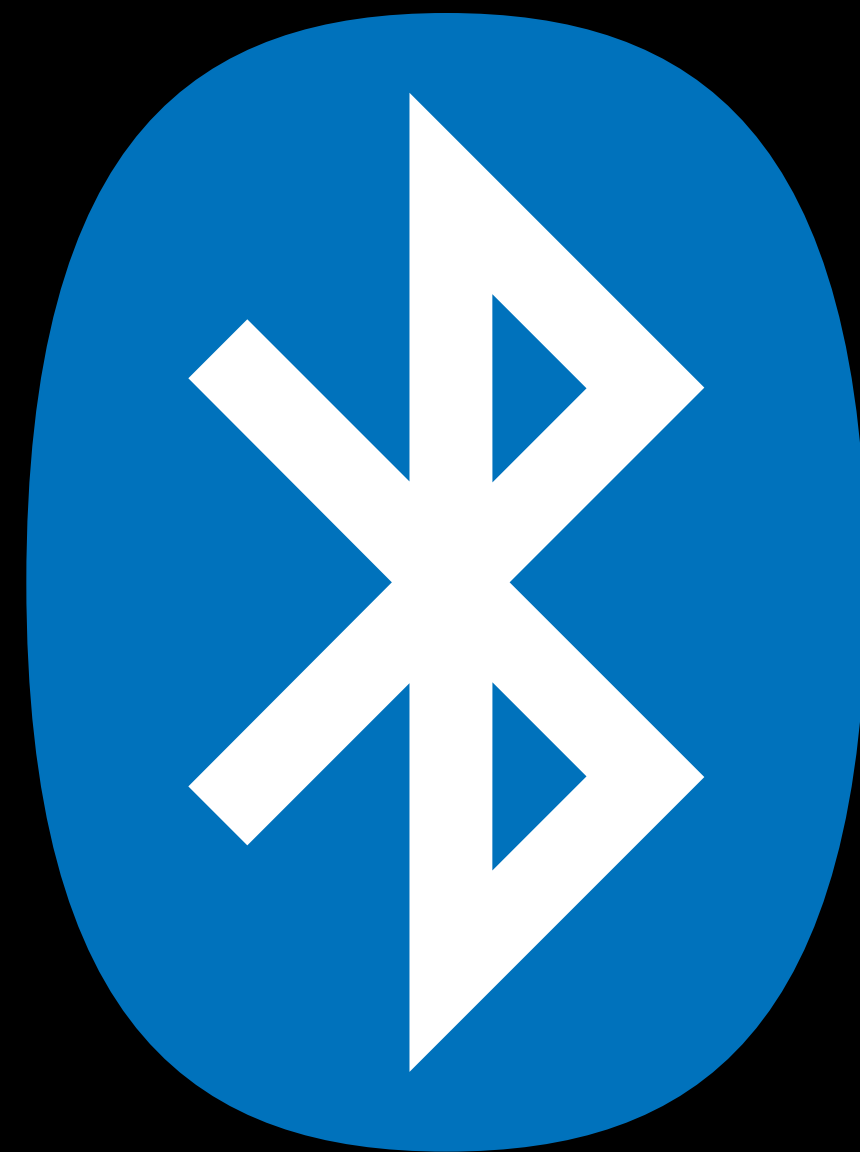
no fancy datatypes, just bytes



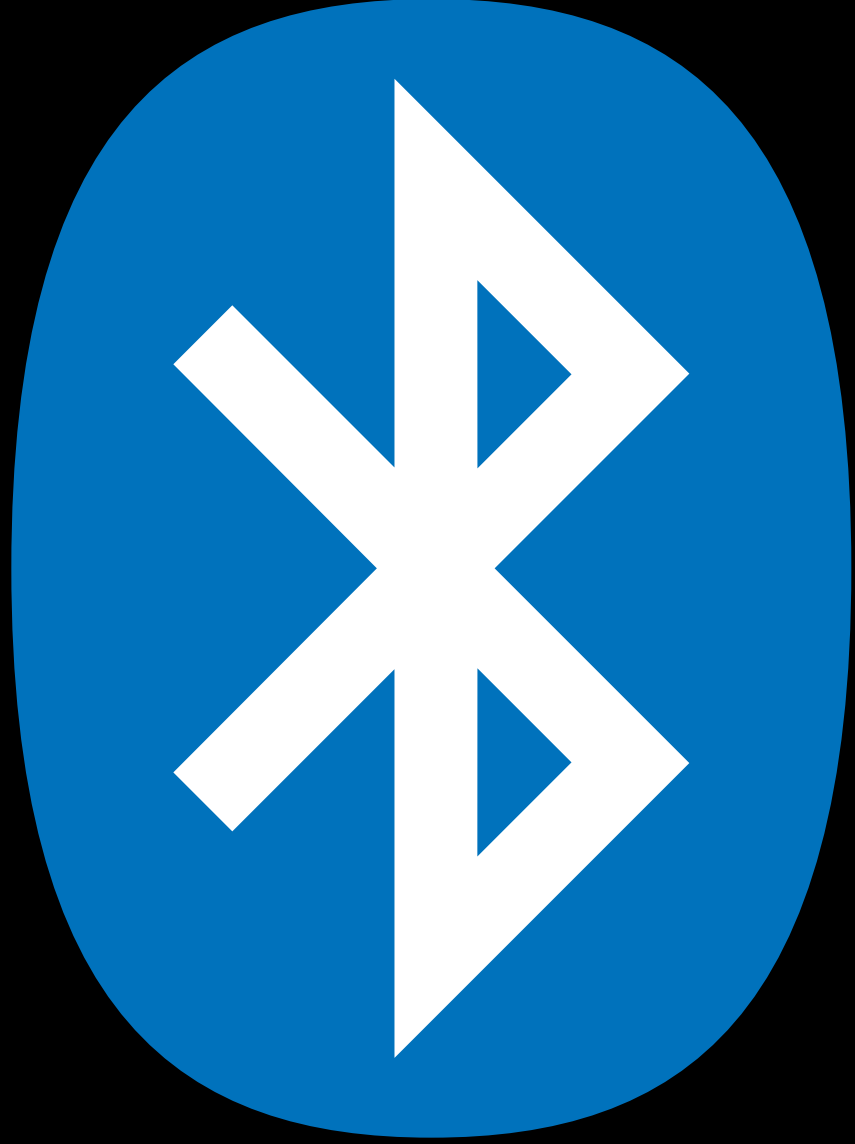
pfew...



boring facts
about
~~fun~~ with



bluetooth

fun with  *bluetooth*

*web
bluetooth.
api*

*still not the fun part
:-)*

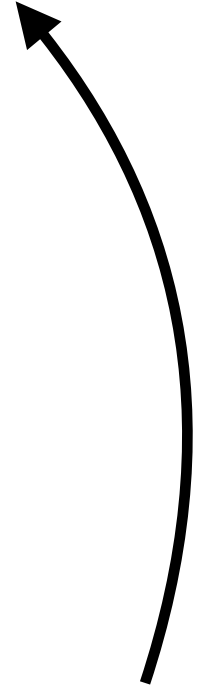


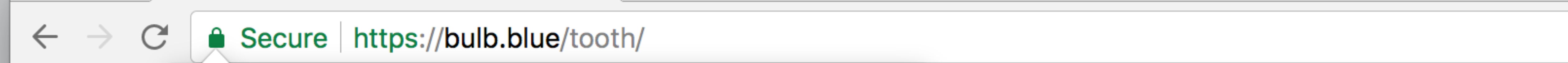
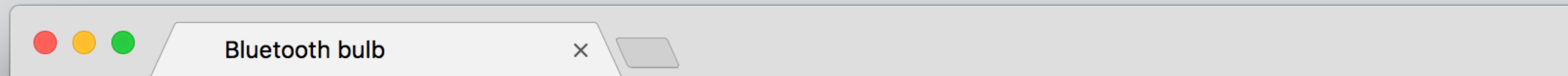
connecting to a device

1

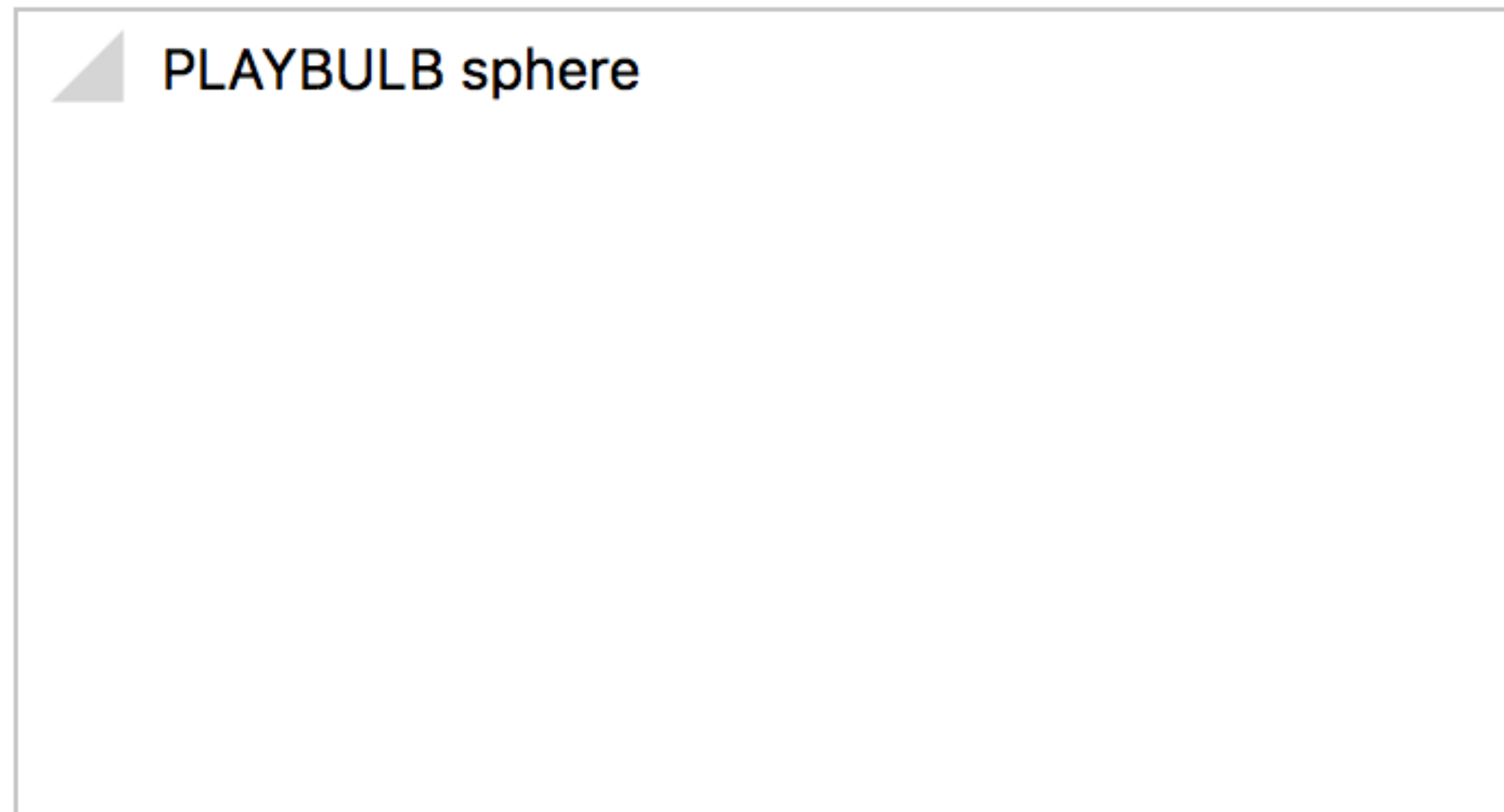
```
navigator.bluetooth.requestDevice({  
  filters: [  
    { namePrefix: 'PLAYBULB' }  
  ],  
  optionalServices: [ 0xff0f ]  
})
```

we tell the browser what
kind of device we want





bulb.blue wants to pair



Cancel

Pair

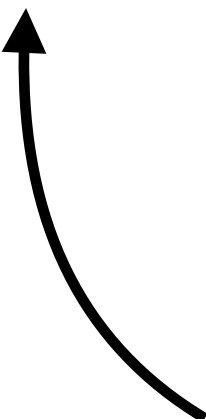
[Get help](#) while scanning for devices...

the user selects
the actual device

2

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

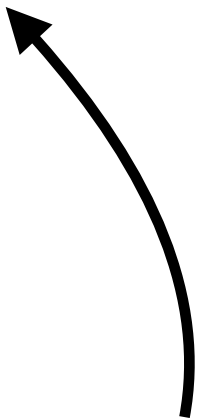
```
.then(device => device.gatt.connect())
```

 connect to the server

3

```
navigator.bluetooth.requestDevice({  
  filters: [  
    { namePrefix: 'PLAYBULB' }  
  ],  
  optionalServices: [ 0xff0f ]  
})  
  
.then(device => device.gatt.connect())  
.then(server => server.getPrimaryService(0xff0f))
```

get the service



4

```
navigator.bluetooth.requestDevice({  
  filters: [  
    { namePrefix: 'PLAYBULB' }  
  ],  
  optionalServices: [ 0xff0f ]  
})  
  
.then(device => device.gatt.connect())  
.then(server => server.getPrimaryService(0xff0f))  
.then(service => service.getCharacteristic(0xfffc))
```

get the characteristic



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

read some bytes



```
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);
    });

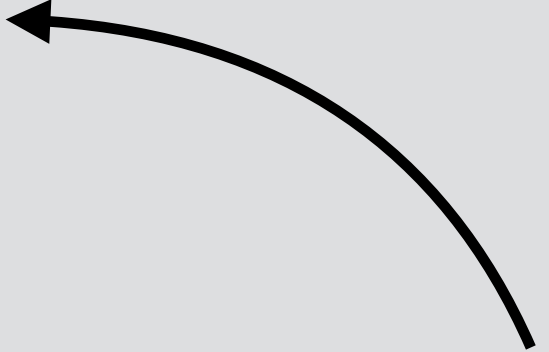
    c.startNotifications();
  })
```



don't forget to start listening

things you need to know:

- the webbluetooth api
- promises
- typed arrays



duh!

browser support



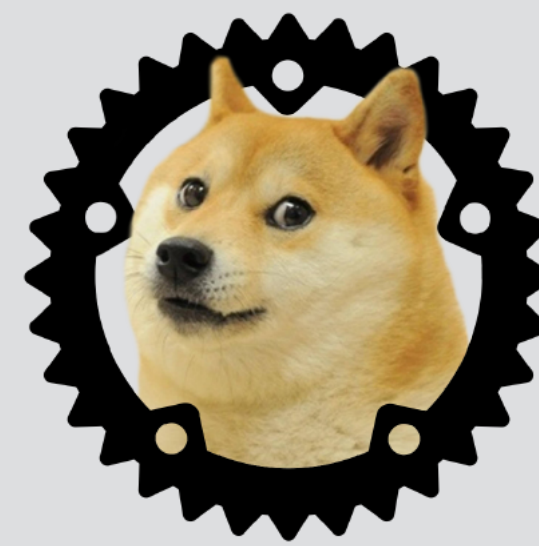
Chrome



Opera



Samsung



Servo
(soon)

browser support



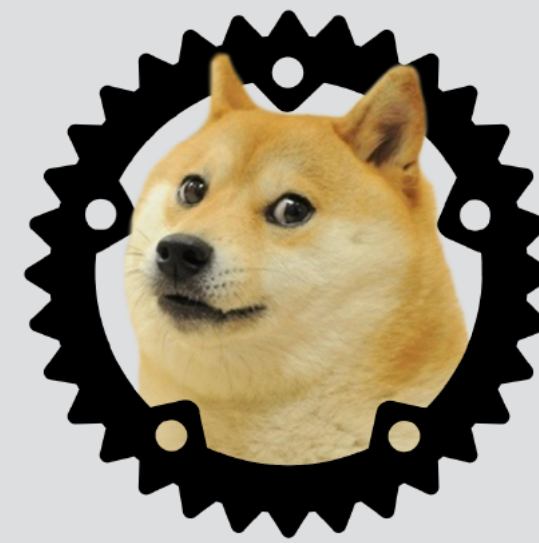
Chrome



Opera



Samsung



Servo
(soon)



Safari

browser support



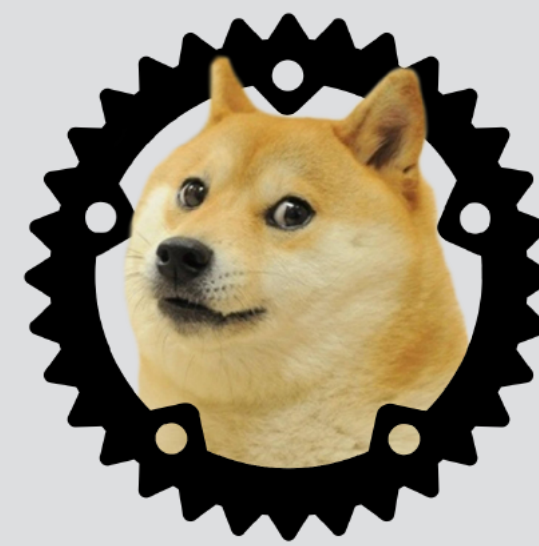
Chrome



Opera



Samsung



Servo
(soon)



Safari

browser support



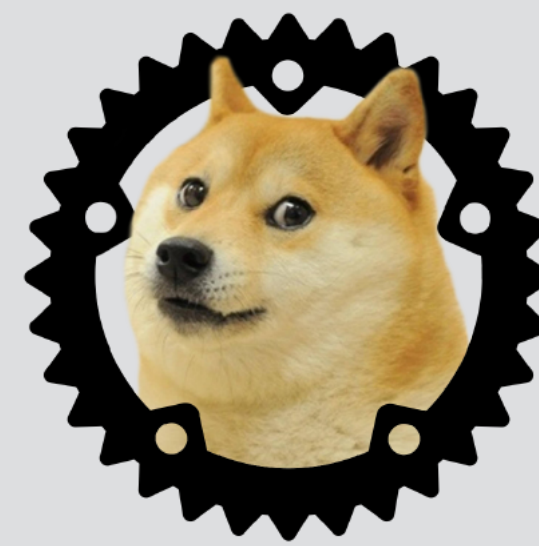
Chrome



Opera



Samsung



Servo
(soon)



WebBLE
for iOS

kinda works

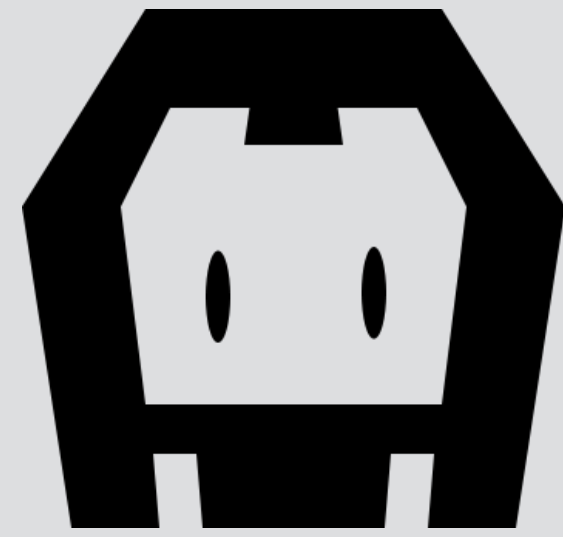
A curved black arrow pointing from the text 'kinda works' down to the WebBLE logo.

and...  ELECTRON



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

and...



APACHE
CORDOVA™

```
npm install cordova-plugin-webbluetooth
```

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

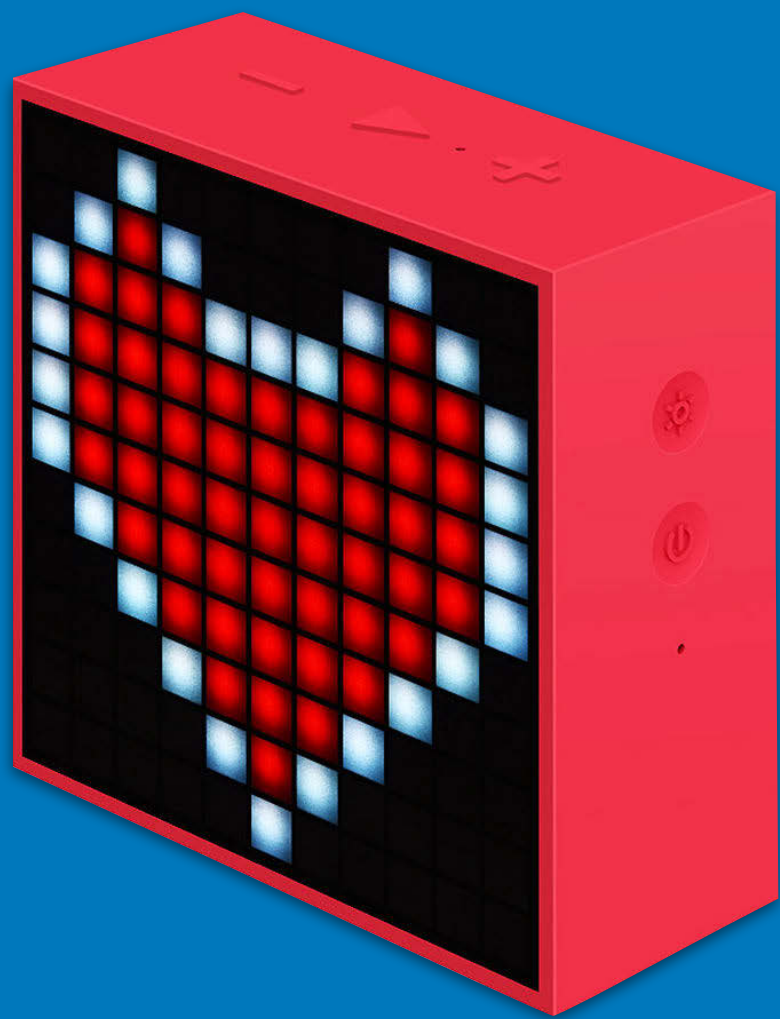
writing a value:

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



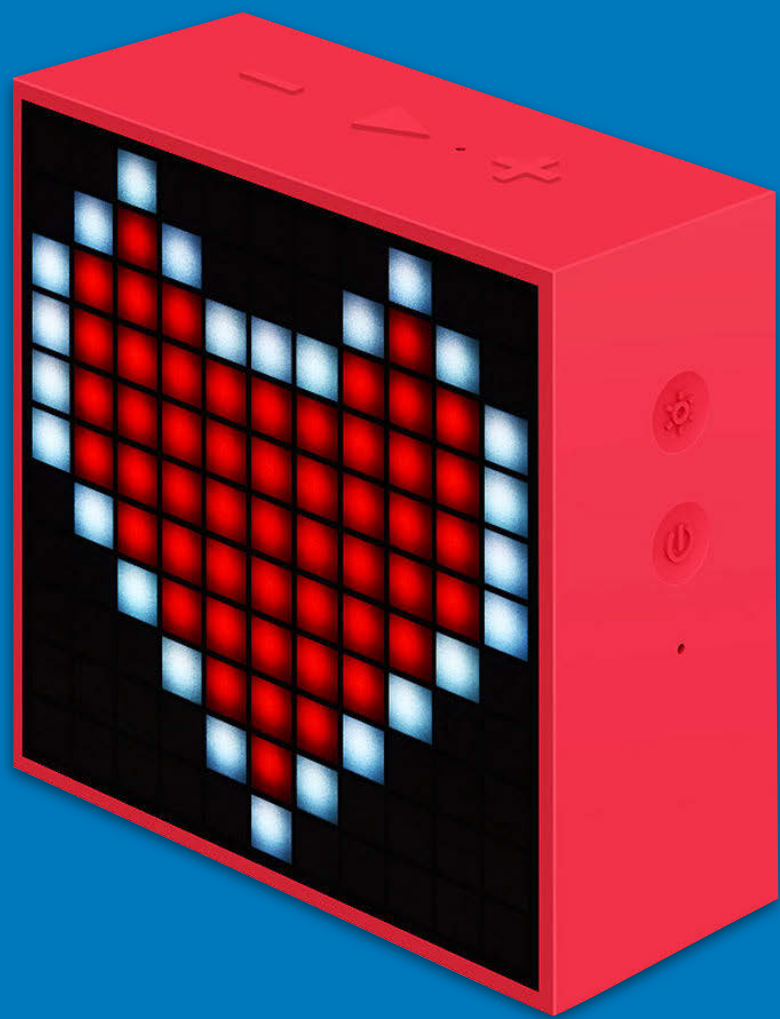
writing a value:

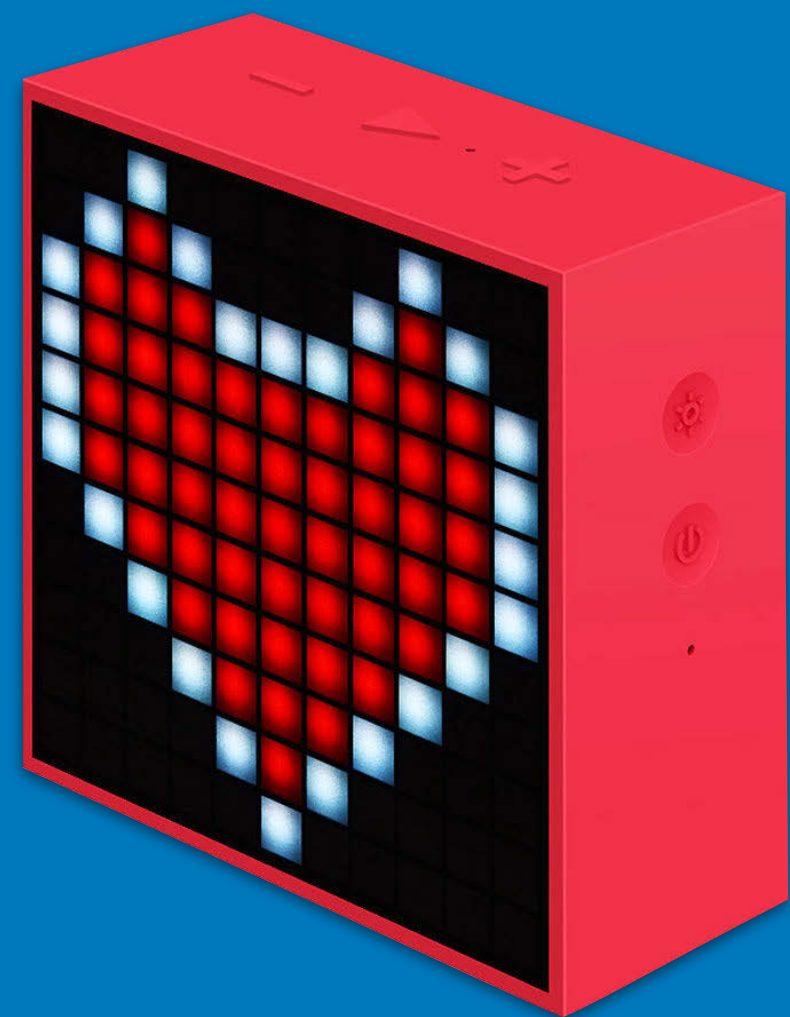
```
function(r, g, b, position) {  
    let buffer = new Uint8Array([  
        0x58, r, g, b, 0x01, position  
    ]);  
  
    ...  
}
```



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] = 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*

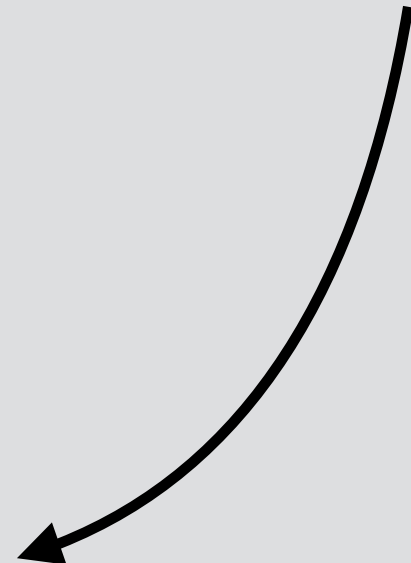
*decompiling
the apk*



don't tell anyone!

finally the fun part

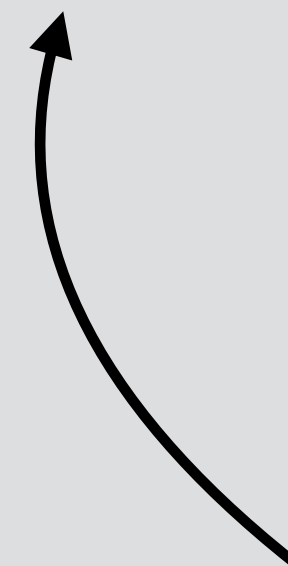
demo





warning

experimental technology



setting low expectations



warning

wifi interference

lowering them even further

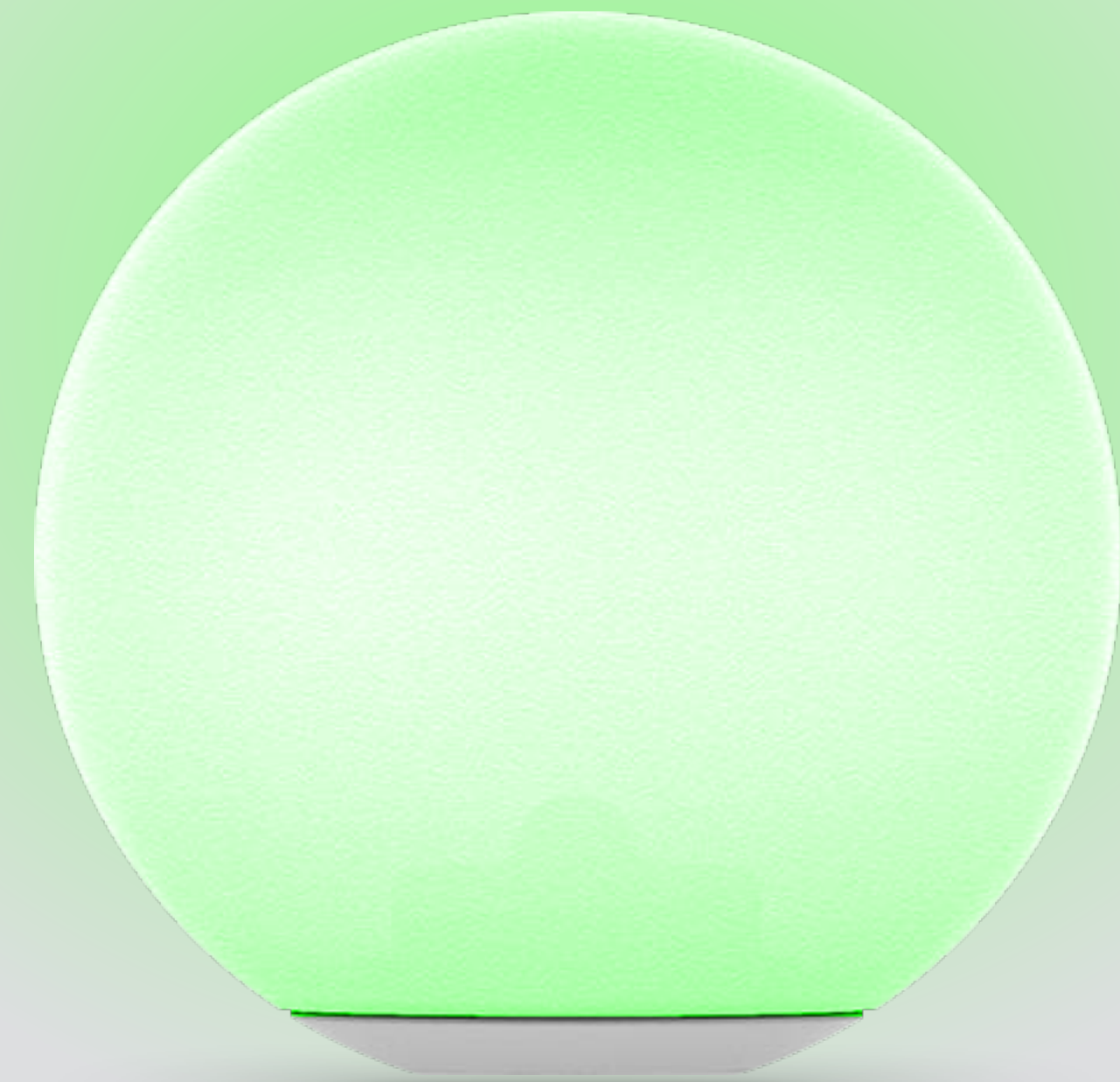


#frontendunited

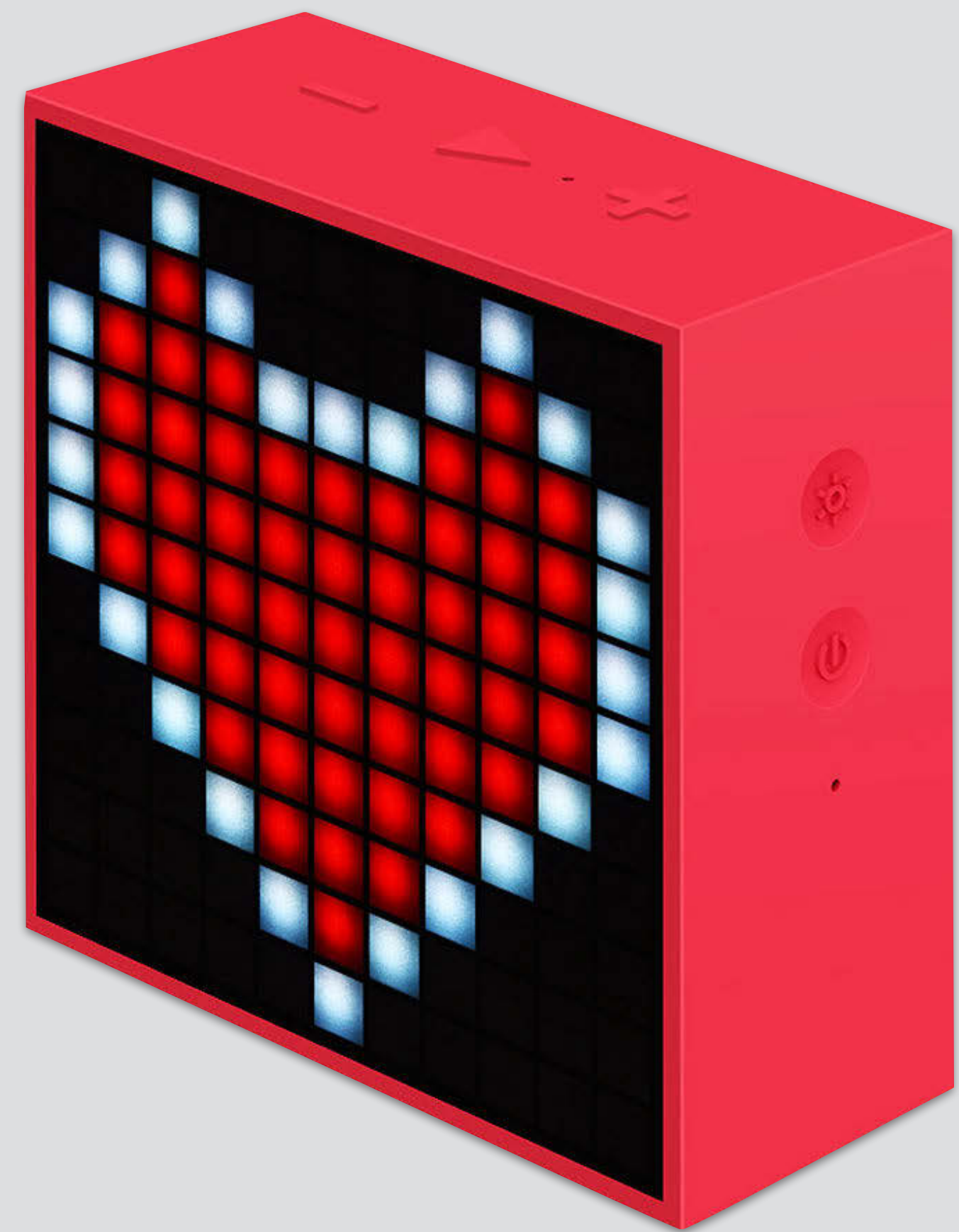
change the colour of a lightbulb

<https://bluetooth.rocks/lightbulb>

<https://github.com/BluetoothRocks/Lightbulb>



*draw pixel art on
a led matrix display*



<https://bluetooth.rocks/pixel>

<https://github.com/BluetoothRocks/Matrix>

*control a lego racer
using a gamepad*

*use css animations to
define a path*



<https://bluetooth.rocks/racer>

<https://github.com/BluetoothRocks/Racer>

*control a drone
from your browser*



<https://bluetooth.rocks/drone>

<https://github.com/BluetoothRocks/Drone>

print on a receipt printer



<https://bluetooth.rocks/printer>

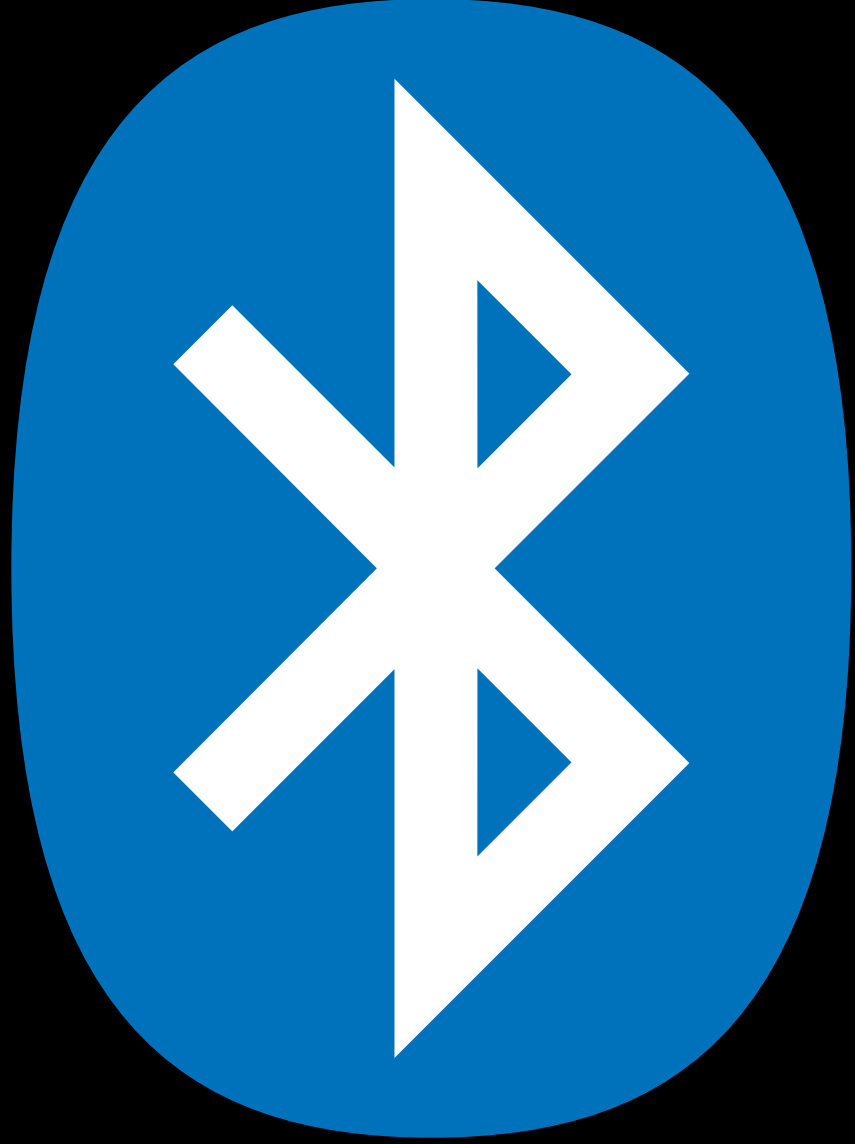
<https://github.com/BluetoothRocks/TweetPrinter>

*find out your
current heartbeat*



<https://bluetooth.rocks/pulse>

<https://github.com/BluetoothRocks/Pulse>

fun with  *bluetooth!*

questions?



@html5test