

An abstract graphic on the left side of the slide, consisting of a network of white lines and small circles on a blue gradient background. The lines are vertical and horizontal, with some diagonal segments, and the circles are of varying sizes, creating a circuit-like or neural network pattern.

SO WHY IS IT BREWSTER TIME?

## PROBLEM

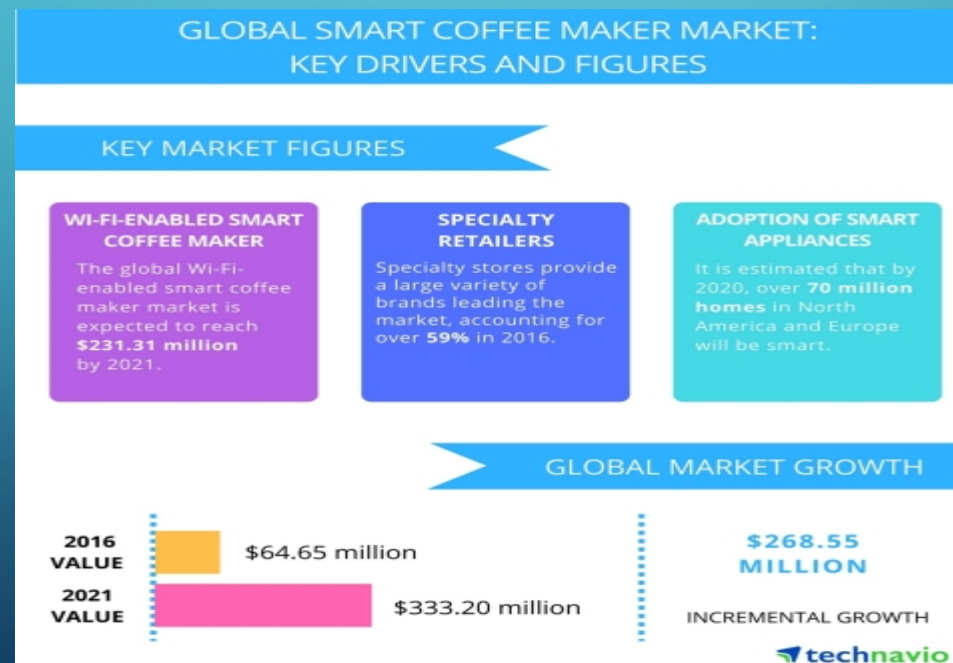
- Differentiation through mobile synchronization and availability of fresh coffee in the workplace setting.

## SOLUTION

- Brewster Time would solve this problem by adding in a social component with real-time automation, notification, and participation. By utilizing the Brewster app, a simple push of a button will initiate Brewster Time to make coffee. Once complete, Brewster Time will notify team members via twitter that fresh coffee is ready and awaiting.

# MARKET RESEARCH

- "Europe, the United States and Japan Account for Over 50% of the World Total Coffee Market."
- "Growing Millennial Population is Fueling Market Growth of Coffee."
- Key Trends





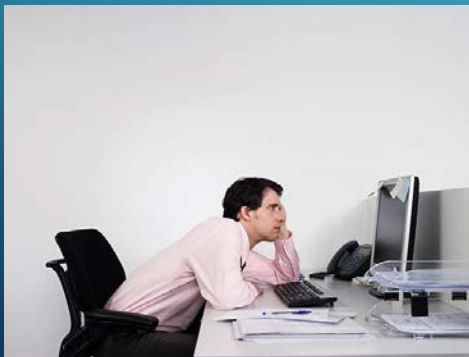
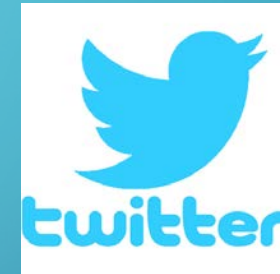
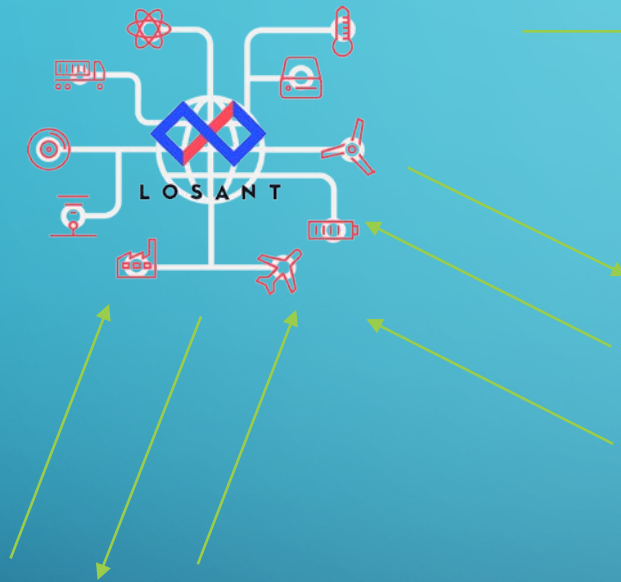
# COMPLETION

- Coffee Shop (premium vs value)
- Corporate Coffee Services (premium vs value)
- Home Brew (premium vs value)

# PLAN – CORPORATE PREMIUM

- Millennials are asking for different office perks (new growing market)
- Employers are see premium coffee as cost effective employee satisfier
- Workplace coffee service internal data value: Customer usage data, Machine Usage, Consumer base channel for advertising
- Corporate coffee service external data value to clients – Productivity, Forecasting/Budgeting, Workplace “differentiator”

# ARCHITECTURE

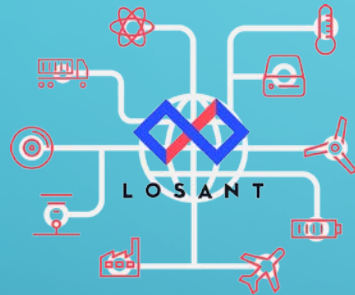




# COMMUNICATION



Mobilejquery  
HTML5  
JavaScript  
Paho MQTT



IoT Platform  
MQTT Broker  
Workflow



Social Media

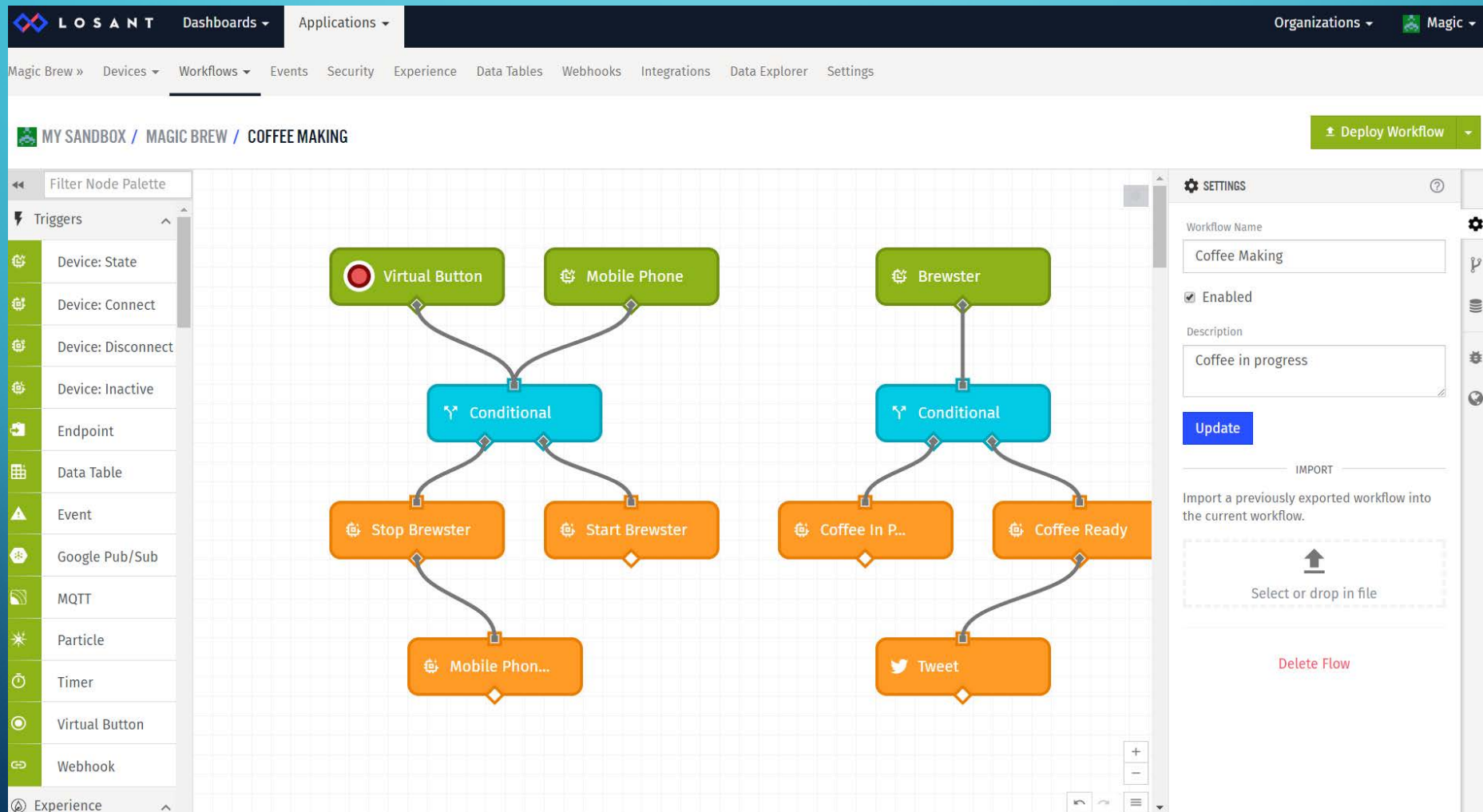


Particle Photon  
ParticleJson  
Structure-MQTT-  
device



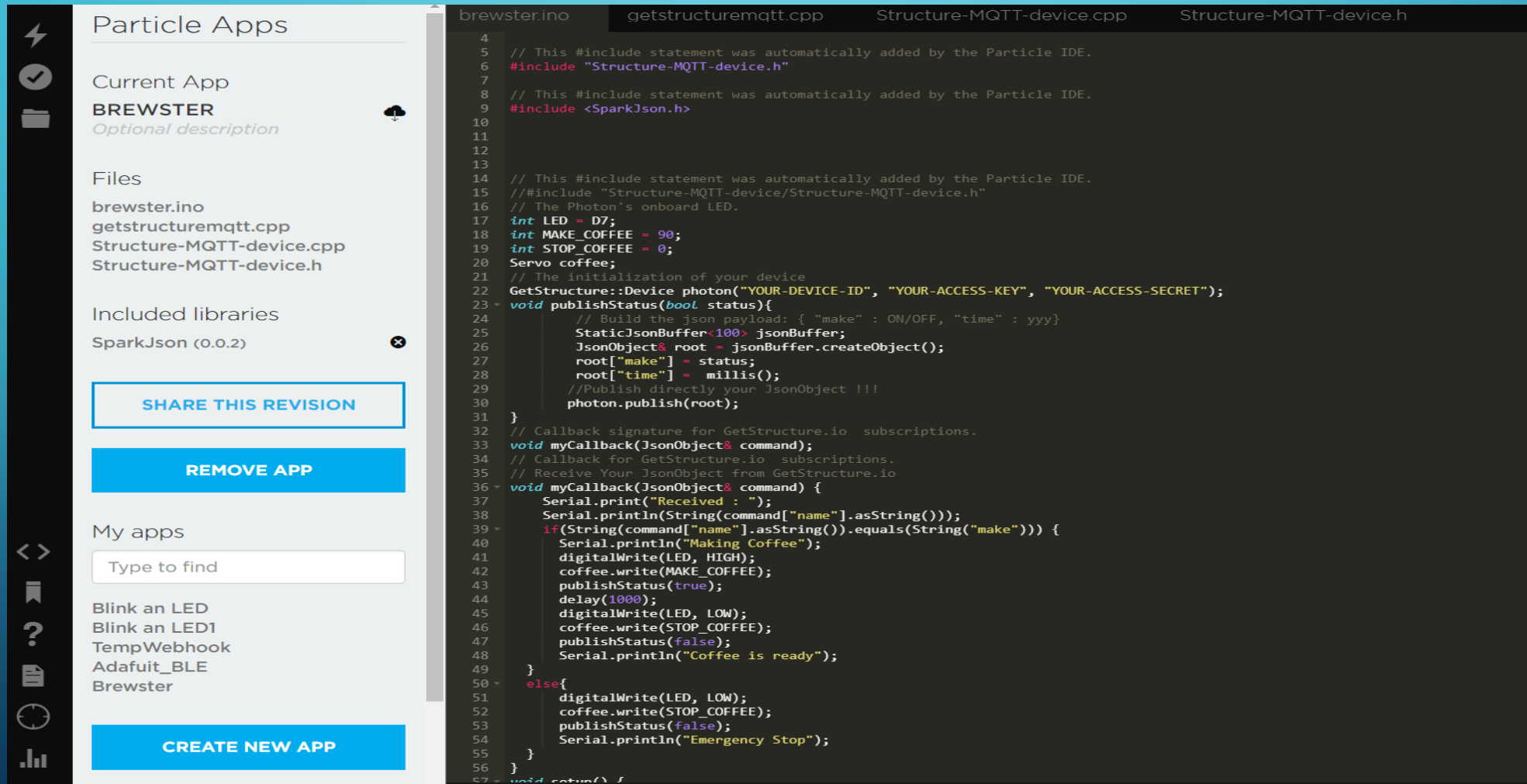
Twitter App

# STRUCTURE APPLICATION





# PARTICLE PHOTON APPLICATION



Particle Apps

Current App

**BREWSTER**

*Optional description*

Files

brewster.ino

getstructuremqtt.cpp

Structure-MQTT-device.cpp

Structure-MQTT-device.h

Included libraries

SparkJson (0.0.2)

SHARE THIS REVISION

REMOVE APP

My apps

Type to find

Blink an LED

Blink an LED1

TempWebhook

Adafruit\_BLE

Brewster

CREATE NEW APP

```
4 // This #include statement was automatically added by the Particle IDE.
5 #include "Structure-MQTT-device.h"
6
7 // This #include statement was automatically added by the Particle IDE.
8 #include <SparkJson.h>
9
10
11
12
13
14 // This #include statement was automatically added by the Particle IDE.
15 // #include "Structure-MQTT-device/Structure-MQTT-device.h"
16 // The Photon's onboard LED.
17 int LED = D7;
18 int MAKE_COFFEE = 90;
19 int STOP_COFFEE = 0;
20 Servo coffee;
21 // The initialization of your device
22 GetStructure::Device photon("YOUR-DEVICE-ID", "YOUR-ACCESS-KEY", "YOUR-ACCESS-SECRET");
23 void publishStatus(bool status){
24     // Build the json payload: { "make" : ON/OFF, "time" : yyy}
25     StaticJsonBuffer<100> jsonBuffer;
26     JsonObject& root = jsonBuffer.createObject();
27     root["make"] = status;
28     root["time"] = millis();
29     // Publish directly your JsonObject !!!
30     photon.publish(root);
31 }
32 // Callback signature for GetStructure.io subscriptions.
33 void myCallback(JsonObject& command);
34 // Callback for GetStructure.io subscriptions.
35 // Receive Your JsonObject from GetStructure.io
36 void myCallback(JsonObject& command) {
37     Serial.print("Received : ");
38     Serial.println(String(command["name"].asString()));
39     if(String(command["name"].asString()).equals(String("make"))){
40         Serial.println("Making Coffee");
41         digitalWrite(LED, HIGH);
42         coffee.write(MAKE_COFFEE);
43         publishStatus(true);
44         delay(1000);
45         digitalWrite(LED, LOW);
46         coffee.write(STOP_COFFEE);
47         publishStatus(false);
48         Serial.println("Coffee is ready");
49     }
50     else{
51         digitalWrite(LED, LOW);
52         coffee.write(STOP_COFFEE);
53         publishStatus(false);
54         Serial.println("Emergency Stop");
55     }
56 }
57 void setup() {
```

# COFFEE MACHINE HACK

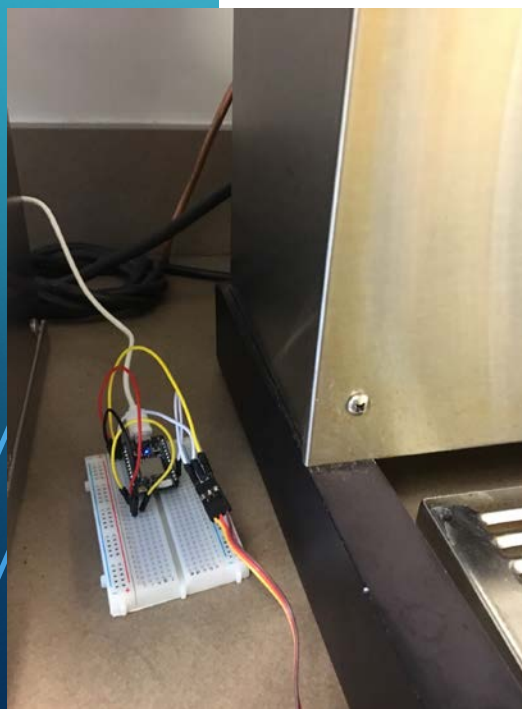
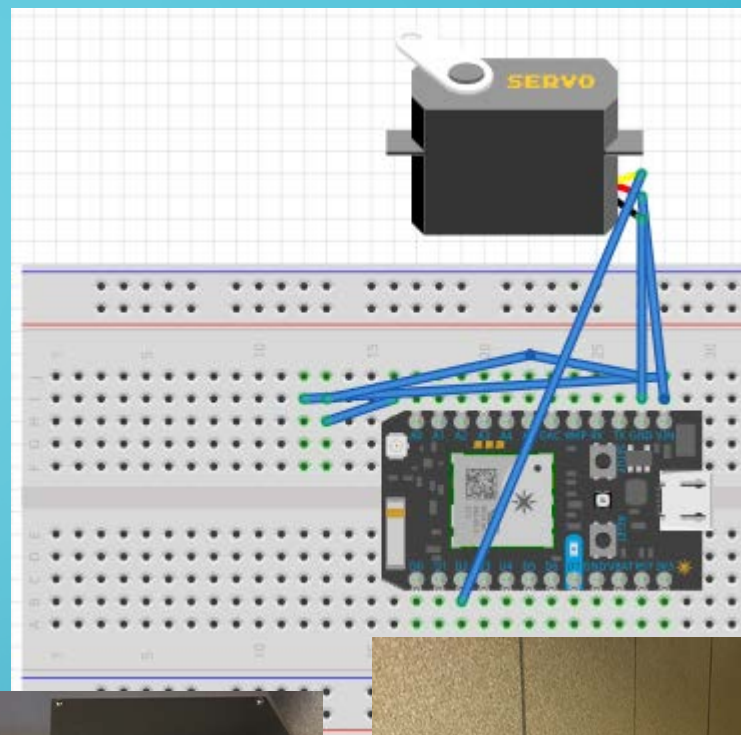
## Coffee Maker

Social coffee break

Make Coffee

Status:

## Received Messages



# IT'S BREWSTER TIME!

- Usefulness
- Technology, Market, and Implementation
- Scalability, Reliability, and Limitations
- Quality of User Experience & Craftsmanship