
VENTURUS4TECH
UMA SALA DE AULA DIFERENTE

Instrutores:

- Formado em Engenharia Elétrica pelo INATEL
- Desenvolvedor C/C++/Android e SW embarcado desde 2001.
- No Venturus desde 2006.
- Contato: pedro.pereira@venturus.org.br

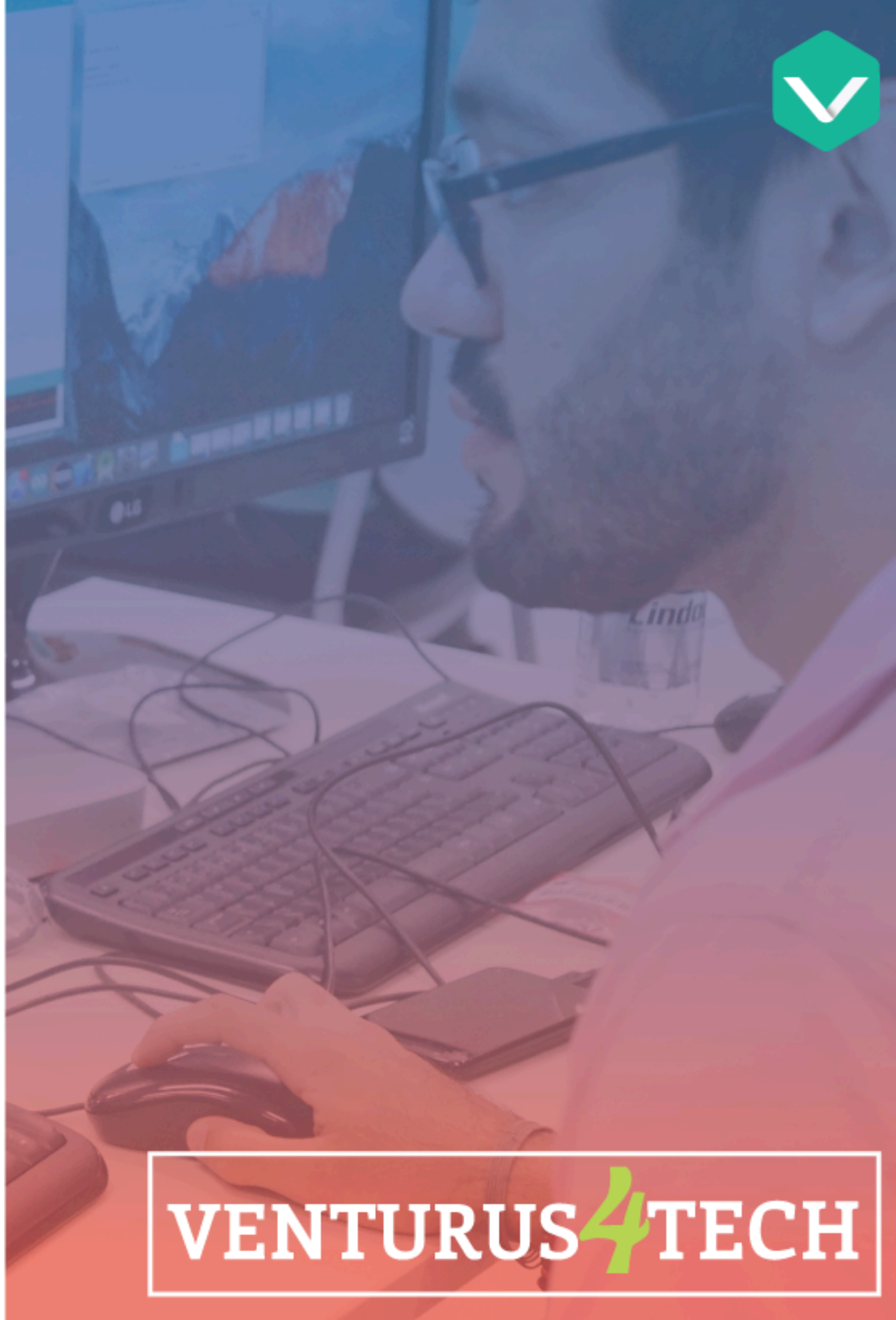
- Formada em Engenharia Elétrica pela PUC-RS, Mestre e Dourota pela USP-SP
- Desenvolvedora C/Java desde 1997.
- No Venturus desde Julho de 2007.
- Contato: liselene.borges@venturus.org.br

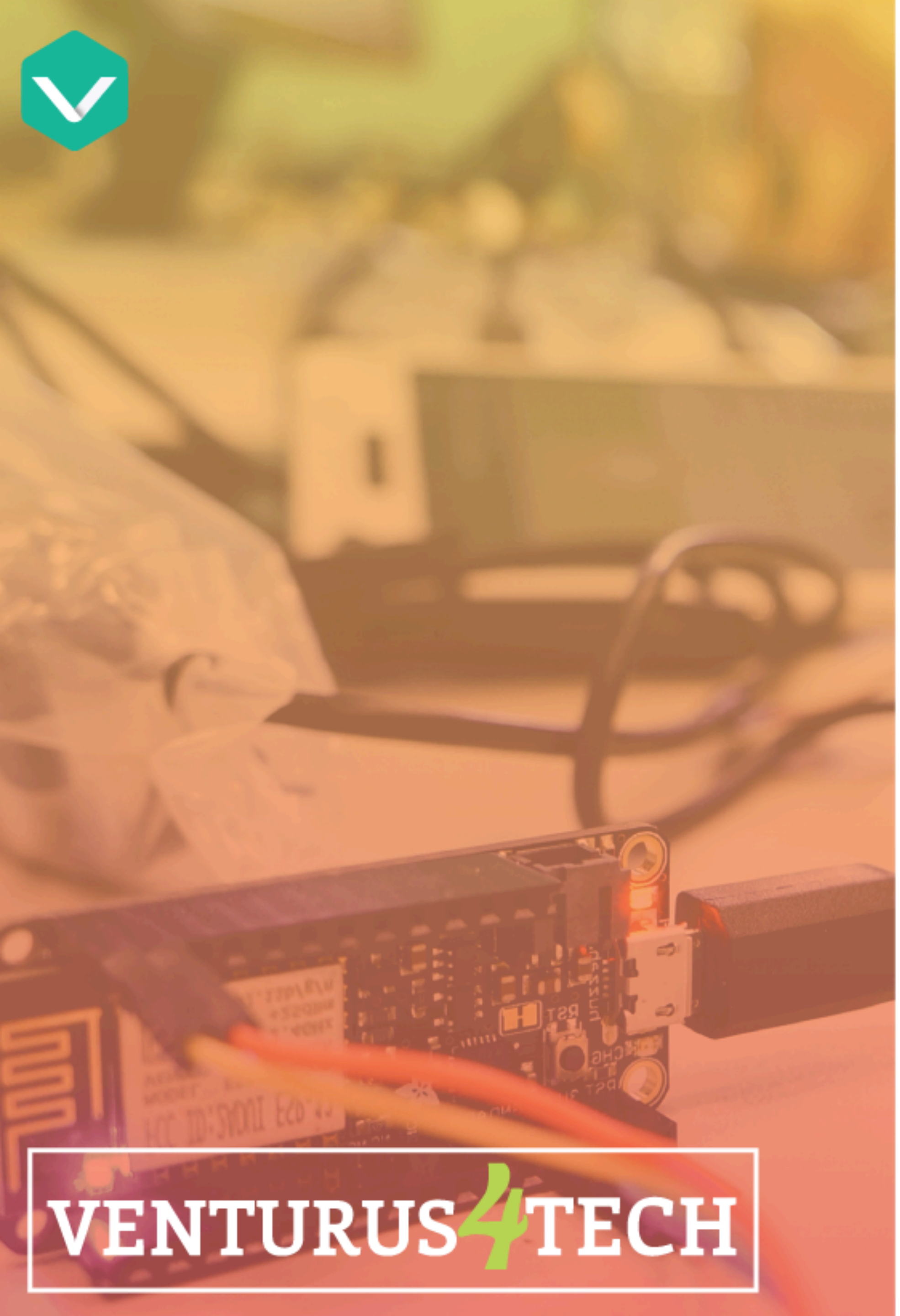
VENTURUS4TECH

Agenda

- Internet das Coisas (IoT)
  - Introdução
  - Aplicações
  - Desafios
    - Segurança
  - Comunicação
  - Protocolos
  - Dispositivos
    - Plataformas
    - Sensores e Atuadores

- Arduino
  - Introdução
  - Placa de desenvolvimento
  - Aplicativo House Control





Introdução IoT

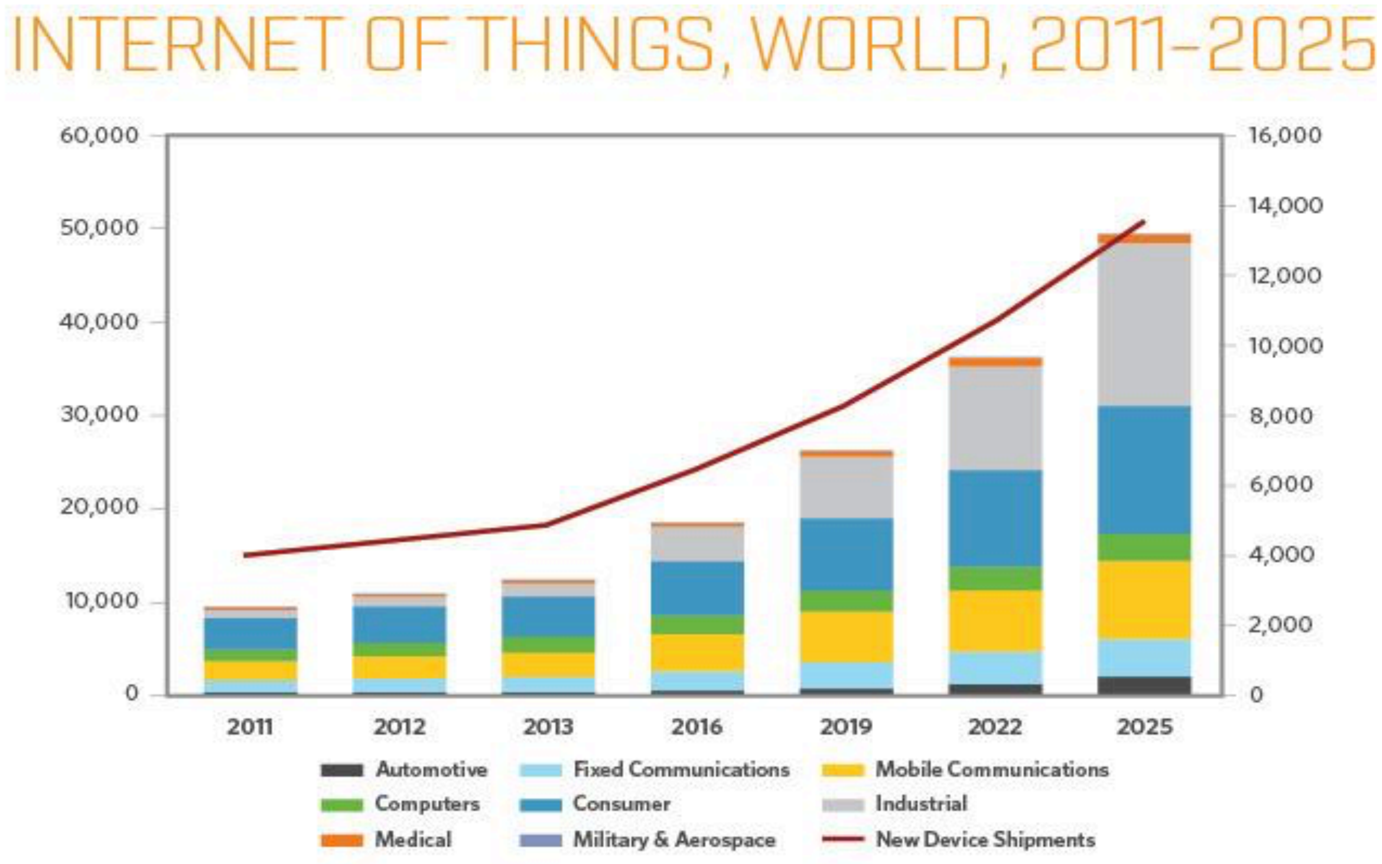
O que é IoT?

É a rede de objetos físicos que possuem tecnologia embarcada para comunicar, captar sinais e interagir consigo mesmos ou com o ambiente externo. (Gartner)

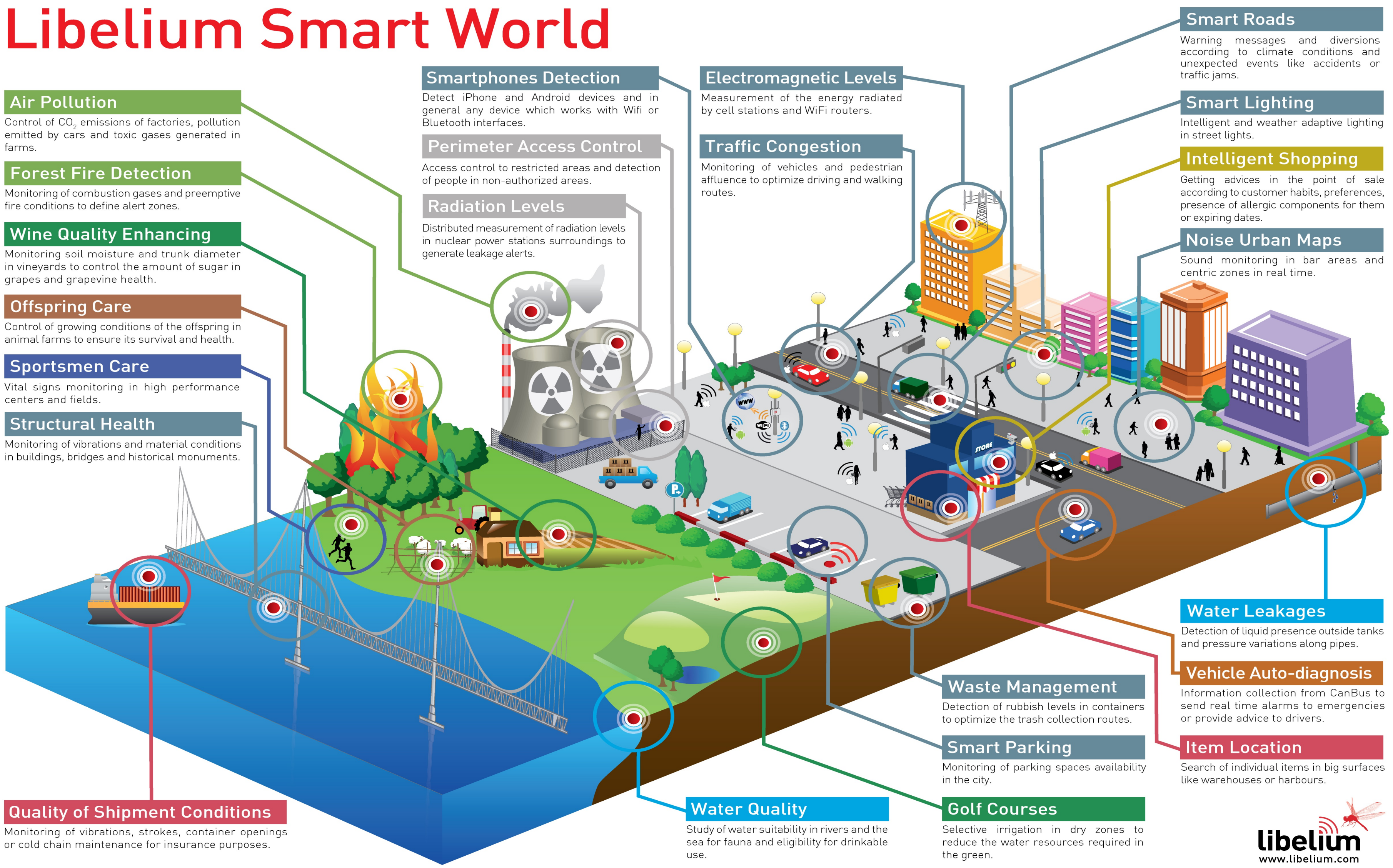
Termo IoT

Criado em 1999 pelo MIT durante necessidade de criar uma rede de devices global (RFID).

Introdução IoT



Source: IHS 2013



VENTURUS4TECH

Desafios

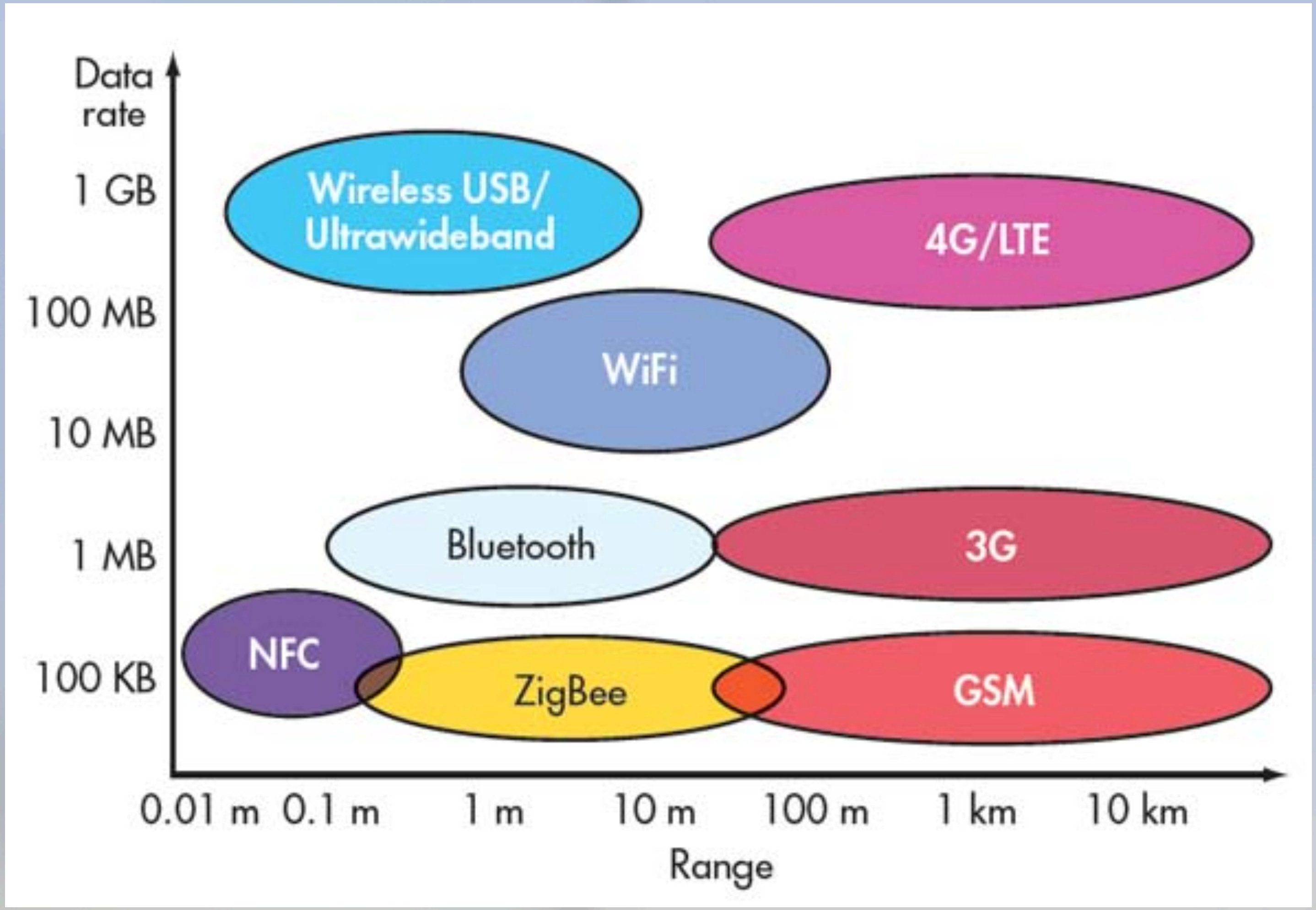
- Custos (Dispositivos, armazenamento, consumo de energia,...)
- Consumo de energia (energias renováveis, novas formas de energia,...)
- Interoperabilidade (Padronização da comunicação)
- Segurança (Controle de acesso aos dispositivos)
- Privacidade (Onde? Quando?)
- Aspectos legais (Coleta e armazenamento dos dados, veículos autônomos)

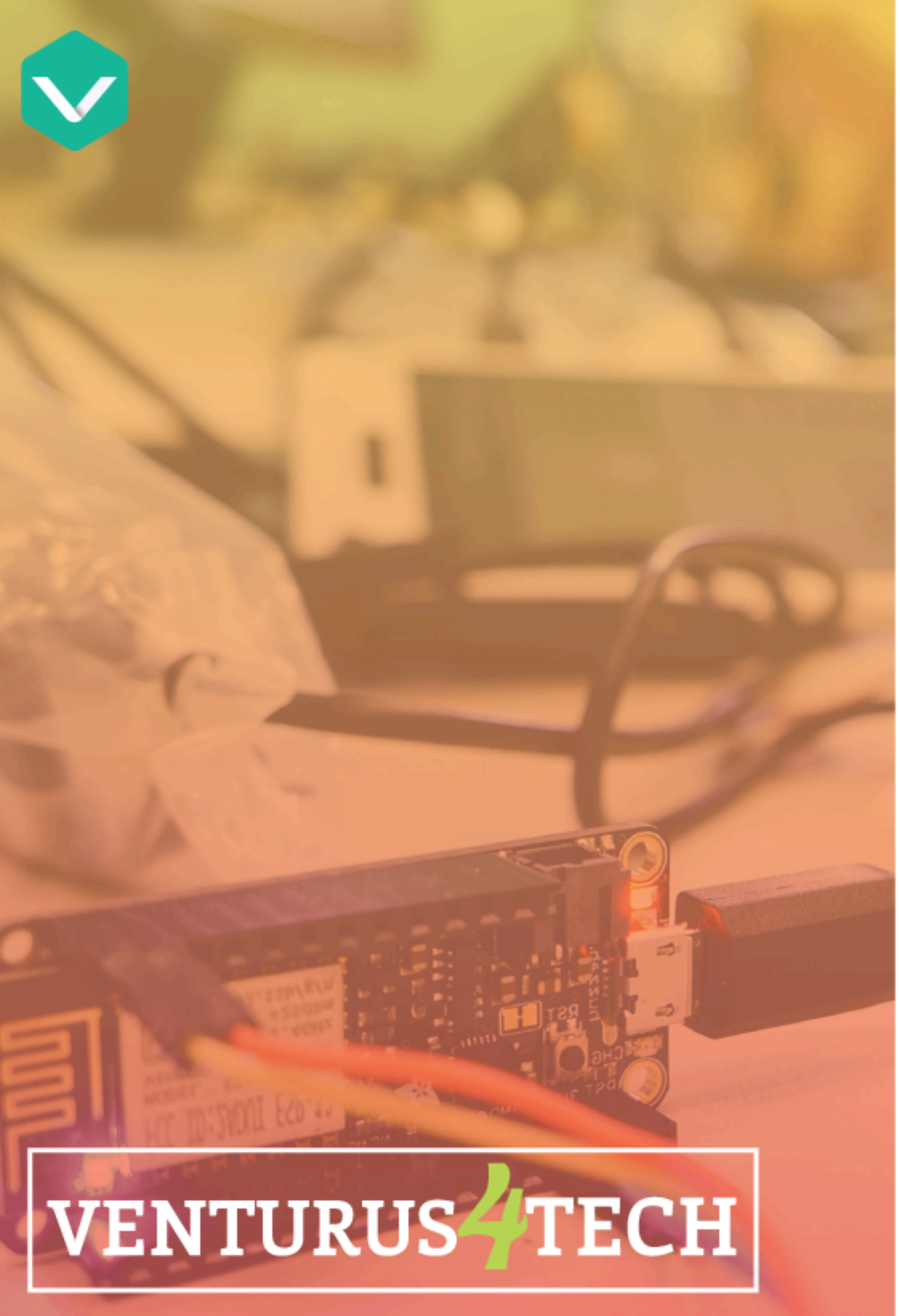
VENTURUS4TECH

Segurança - exemplos

- TV LG
  - Caso de roubo de senha
- Lâmpada Philips Hue
  - Caso do hacker que fez a luz ficar piscando
- Babá eletrônica
  - Espionagem
- Carro conectado
  - Qualquer um sabendo sua rotina e localização

Comunicação





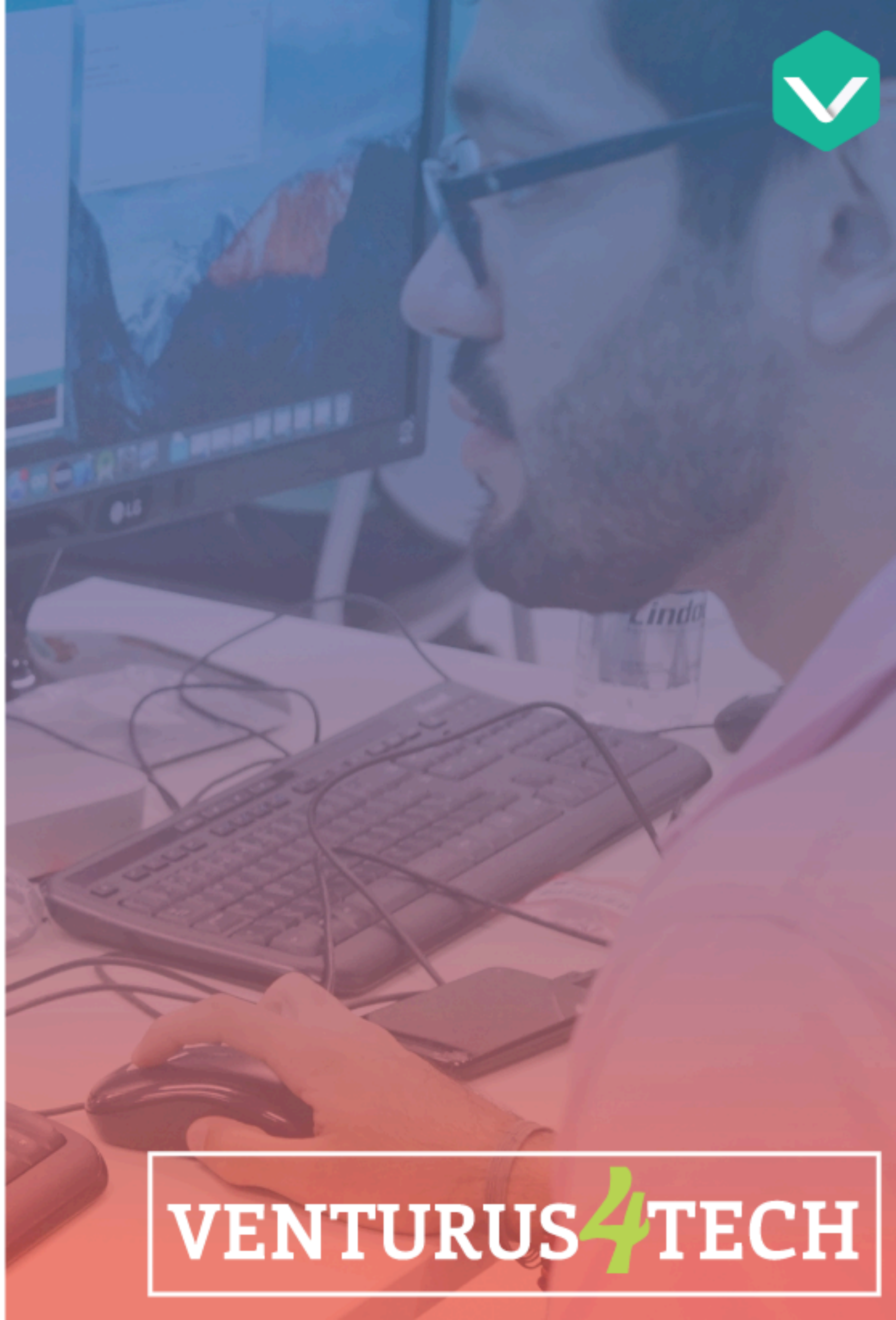
Protocolos IoT

Principais protocolos de comunicação

- HTTP (HyperText Transfer Protocol)
- CoAP (Constrained Application Protocol)
- AMQP (Advanced Message Queuing Protocol)
- XMPP (eXtensible Messaging and Presence Protocol)
- MQTT (Message Queue Telemetry Transport)

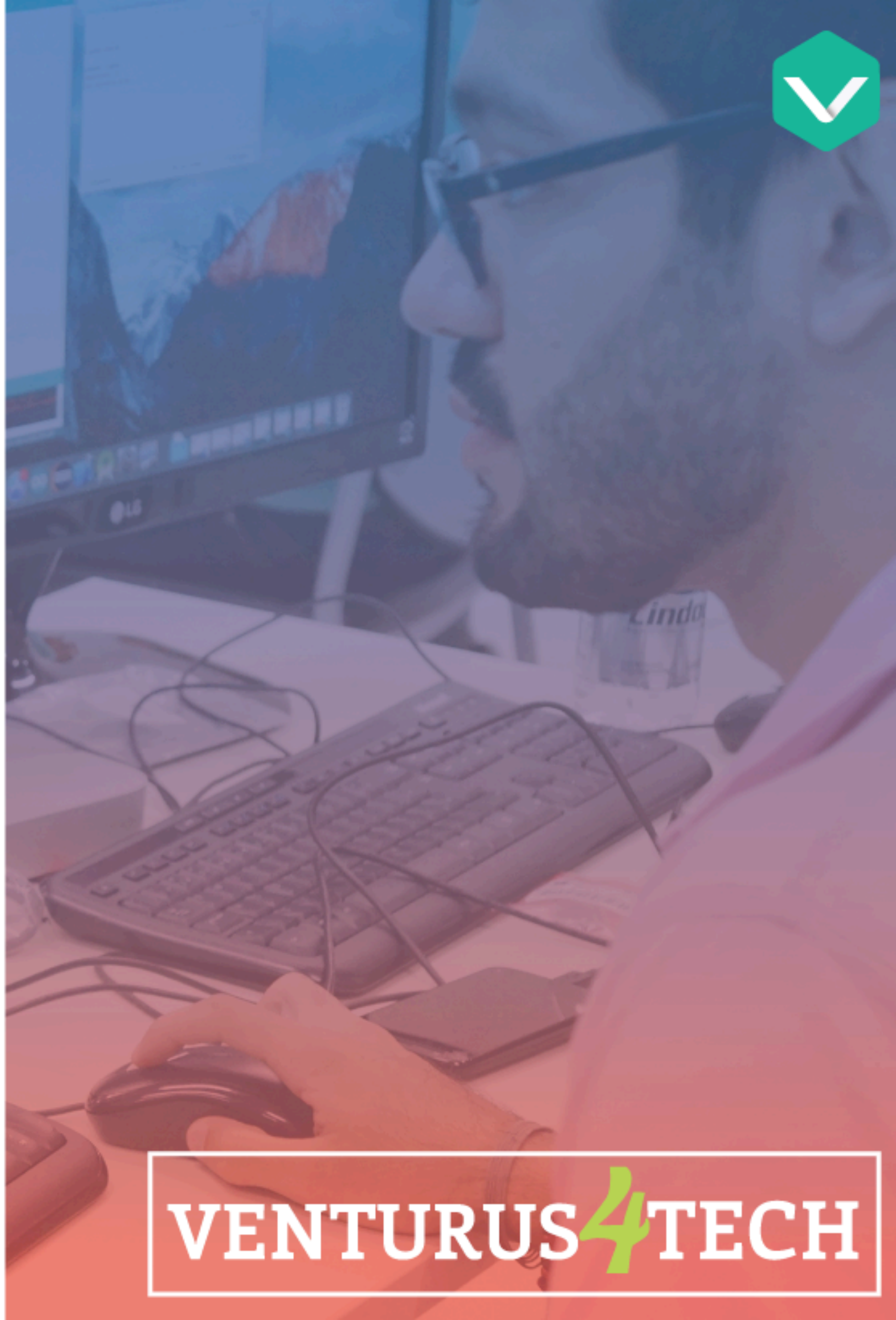
Protocolos IoT - MQTT

- Message Queue Telemetry Transport
  - Criado pela IBM/Arcom em 1999.
  - Publish/Subscribe
  - Simples e leve (poucos kb)
  - Minimiza uso de banda/ internet e recursos
  - Possui QoS (garantia de mensagem entregue)
  - Precisa de um Broker para funcionar

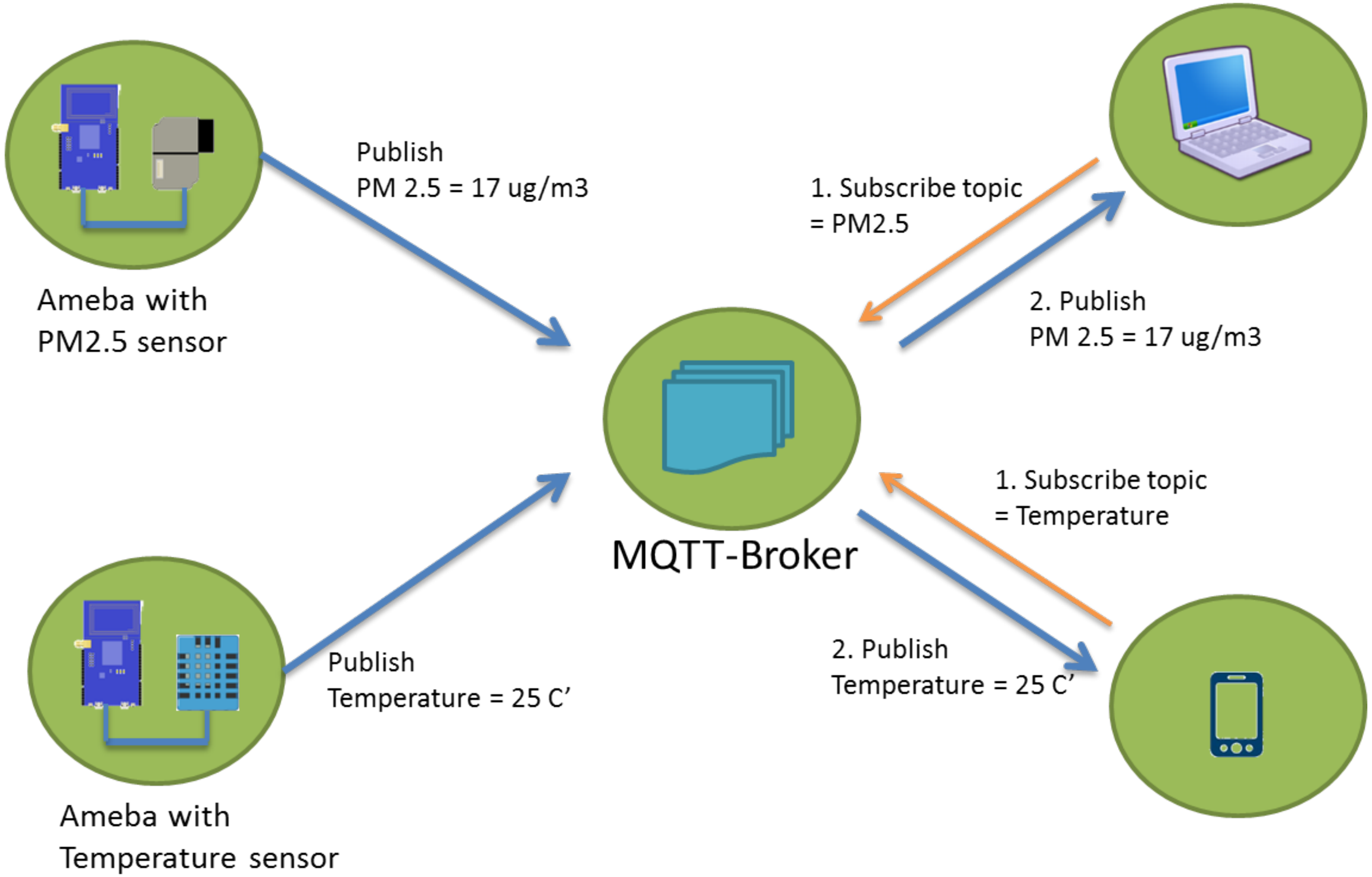


Protocolos IoT - MQTT

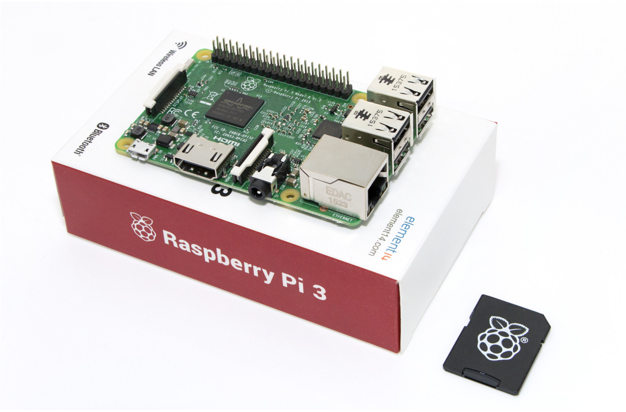
- Comparado com HTTPS, MQTT:
  - 93 vezes mais eficiente (mais mensagens por intervalo de tempo)
  - 11 vezes menos bateria para transmissão
  - 170 vezes menos para recepção.
  - Garantia de entrega
  - Armazena mensagens não lidas
  - Last will & testament

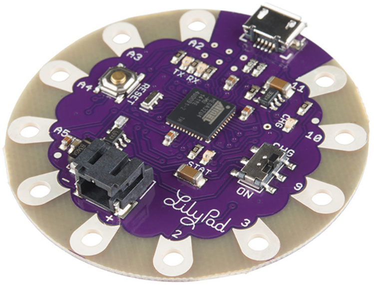


Protocolos IoT - MQTT



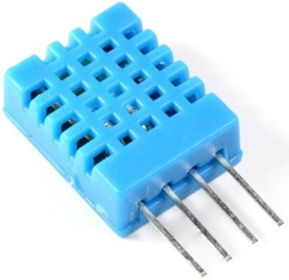
Dispositivos IoT - plataformas



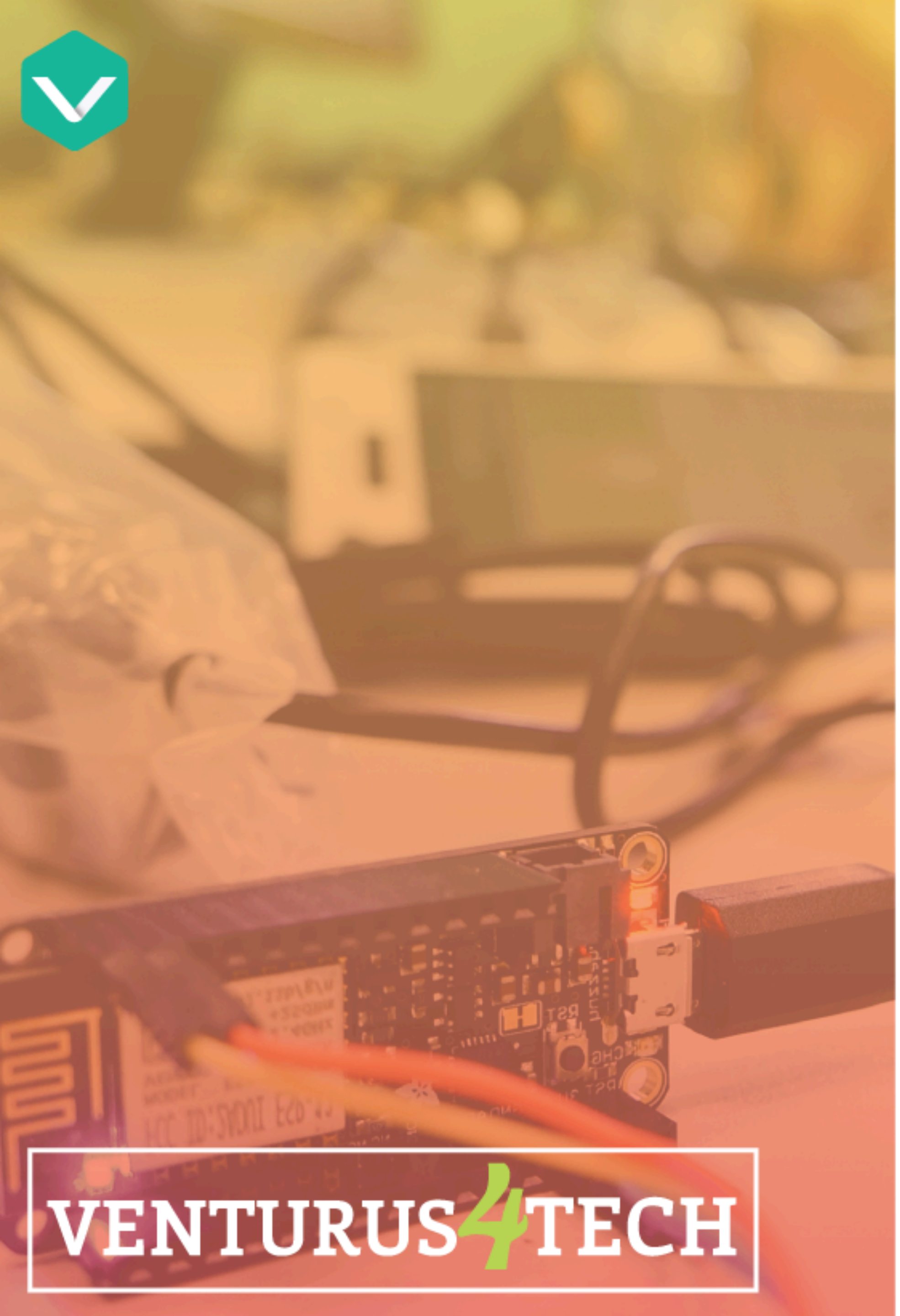


VENTURUS4TECH

Dispositivos IoT – sensors/atuadores



VENTURUS4TECH



Arduino

O que é?

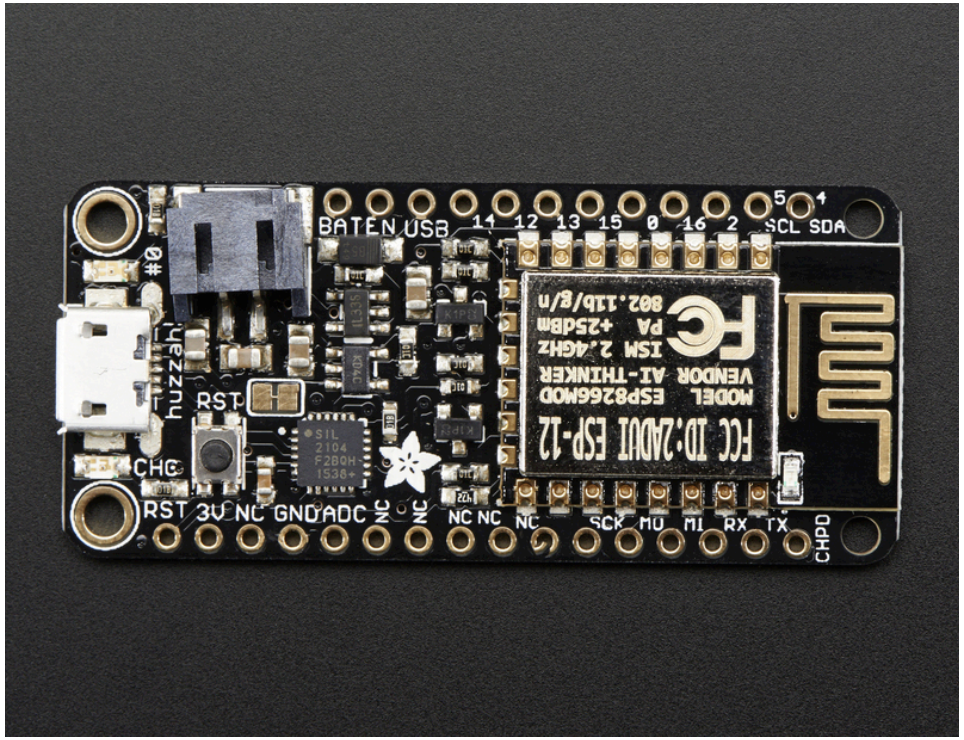
- Hardware
- Ambiente de desenvolvimento

Vantagens

- Prototipagem
- Open source (hardware e software)
- Comunidade (wiki, exemplos, dúvidas)

Placas do Curso - ESP8266

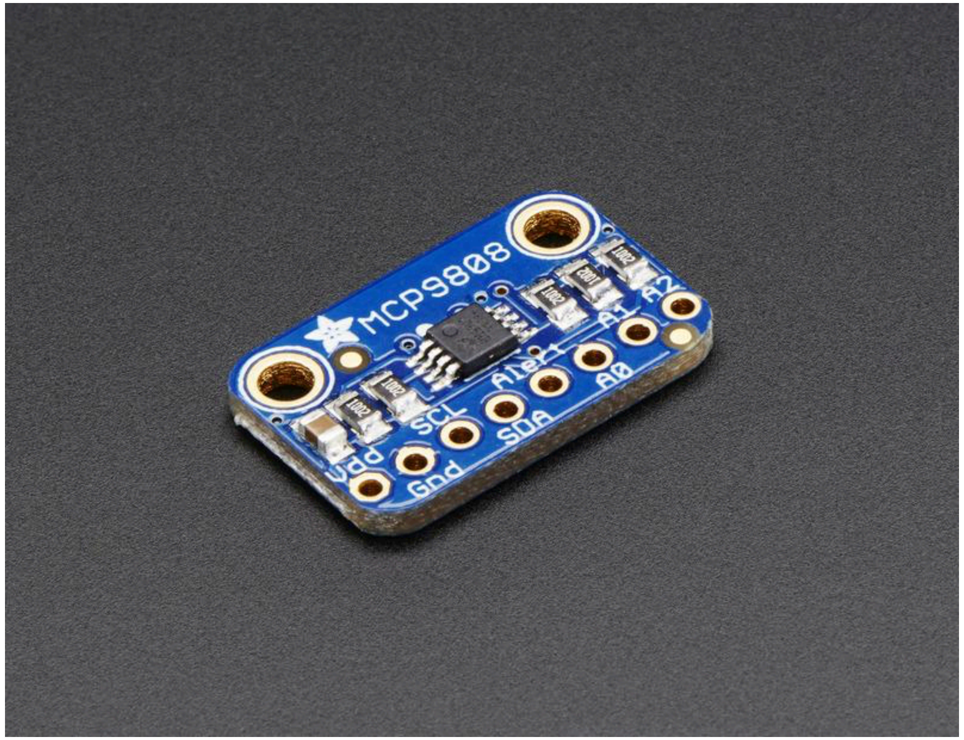
- Light as a (large?) feather - 6 grams
- ESP8266 @ 40MHz with 3.3V logic/power
- 4MB of FLASH (32 MBit)
- 3.3V regulator with 500mA peak current output
- CP2104 USB-Serial converter onboard with 921600 max baudrate
- Auto-reset support for getting into bootload mode before firmware upload
- 9 GPIO pins - can also be used as I2C and SPI
- 1 x analog inputs 1.0V max
- Built in 100mA lipoly charger with charging status indicator LED
- Pin #0 red LED for general purpose blinking. Pin #2 blue LED for bootloading debug & general purpose blinking
- Power/enable pin
- Reset button



VENTURUS4TECH

Placas do Curso - MCP9808

- Simple I2C control
- Up to 8 on a single I2C bus with adjustable address pins
- 0.25°C typical precision over -40°C to 125°C range (0.5°C guaranteed max from -20°C to 100°C)
- 0.0625°C resolution
- 2.7V to 5.5V power and logic voltage range
- Operating Current: 200 μA (typical)

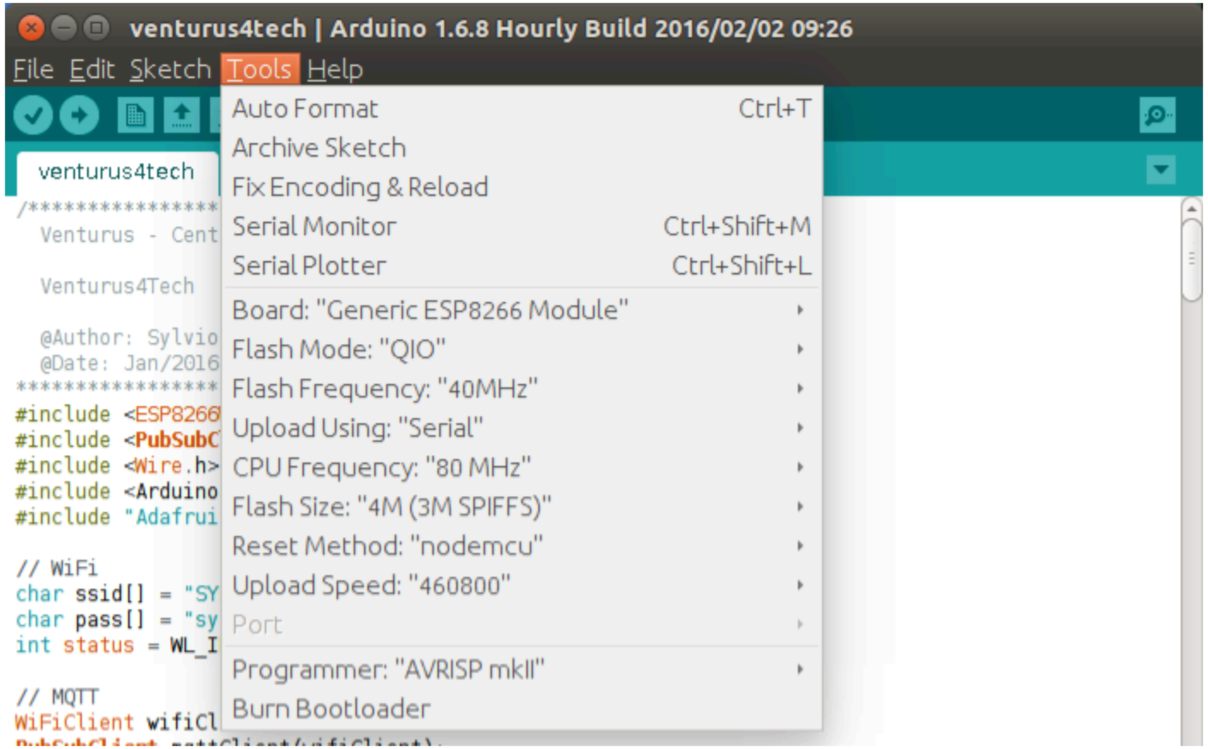


Preparando o ambiente (MAC)

- Instalar FTDI Driver:
  - https://goo.gl/lXSTRO
- Adicionar ESP8266 board package:
  - http://arduino.esp8266.com/stable/package_esp8266com_index.json
- Instalar board: Menu Tools/Board/Boards Manager
  - Instalar esp8266
- Adicionar Libraries: Menu Sketch/Include Library/Manage Libraries
  - PubSubClient
  - ArduinoJson
  - Adafruit MCP9808

Configurando a placa ESP8266

- Port:
  - usbserial OU
  - usbmodem



Exemplo 1 - Piscar LED

```
// Setup é executada apenas na inicialização
void setup() {
  pinMode(0, OUTPUT); // configura porta 0 como saída
}

// Função executada sem parar
void loop() {
  digitalWrite(0, LOW); // liga o LED
  delay(1000);          // aguarda 1 segundo
  digitalWrite(0, HIGH);// desliga o LED
  delay(1000);          // aguarda 1 segundo
}
```

VENTURUS4TECH

Exemplo 2 - Piscar LED sem bloqueio

```
int ledState = LOW;

unsigned long previousMillis = 0;
const long interval = 1000;

void setup() {
  pinMode(0, OUTPUT);
}
```

```
void loop() {
  unsigned long currentMillis = millis();
  if(currentMillis - previousMillis >= interval) {
    previousMillis = currentMillis;
    if (ledState == LOW) {
      ledState = HIGH;
    } else {
      ledState = LOW;
    }
    digitalWrite(0, ledState);
  }
}
```

Exemplo 3 - Porta Serial

```
int ledState = LOW;

unsigned long previousMillis = 0;
const long interval = 1000;

void setup() {
  Serial.begin(115200);
  pinMode(0, OUTPUT);
}
```

```
void loop() {
  unsigned long currentMillis = millis();
  if(currentMillis - previousMillis >= interval) {
    previousMillis = currentMillis;
    if (ledState == LOW) {
      Serial.println(“Desligou LED”);
      ledState = HIGH;
    } else {
      Serial.println(“Ligou LED”);
      ledState = LOW;
    }
    digitalWrite(0, ledState);
  }
}
```

Exemplo 4 - WiFi

```
#include <ESP8266WiFi.h>

// WiFi
char ssid[] = "WSony_Lab";    // Nome da rede WiFi
char pass[] = "Wsony2016";    // Senha

// Declarando funções do código
void setup_wifi();

void setup() {
  pinMode(0, OUTPUT);
  setup_wifi();
}
```

```
void setup_wifi() {
  Serial.println();
  Serial.print("Conectando ao SSID: ");
  Serial.println(ssid);

  WiFi.begin(ssid, pass);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
  Serial.println("Conectado ao WiFi!");
  Serial.print("Endereço IP: ");
  Serial.println(WiFi.localIP());
}
```

Exemplo 4 - WiFi + LED Azul ligado

```
#include <ESP8266WiFi.h>

// WiFi
char ssid[] = "WSony_Lab";    // Nome da rede WiFi
char pass[] = "Wsony2016";    // Senha

// Declarando funções do código
void setup_wifi();

void setup() {
  pinMode(0, OUTPUT);
  pinMode(2, OUTPUT);
  setup_wifi();
}
```

```
void setup_wifi() {
  (...)

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    digitalWrite(2, !digitalRead(2));
    Serial.print(".");
  }

  digitalWrite(2, LOW);
  (...)
}
```