



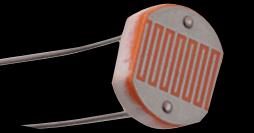
Projeto 02

Coisas Replicadas – Resumo da Ópera

Jan K. S. – janks@puc-rio.br

ENG4051 – Projeto Internet das Coisas

Sensor de Luz



```
int leitura = analogRead(pino);  
int porcentagemLuz = map(leitura, 0, 4095, 0, 100);
```

Millis

```
unsigned long instanteAnterior = 0;  
  
void loop () {  
    unsigned long instanteAtual = millis();  
    if (instanteAtual > instanteAnterior + 1000) {  
        Serial.println("+1 segundo");  
        instanteAnterior = instanteAtual;  
    }  
}
```

String

```
String texto1 = "Olá, mundo!";  
int numero = 100 * 2;  
String texto2 = String(numero);  
int numero2 = texto2.toInt() + 42;  
  
String texto3 = "aaa" + texto2;
```

```
bool ehIgual = texto2 == texto3;  
bool comecaComOla = texto1.startsWith("Olá");
```

```
char caracter = texto1[2]; // 'á'  
int totalCaracteres = texto1.length(); // 11
```

```
String trecho = texto1.substring(0, 3); // "Olá"  
String trechoFinal = texto1.substring(5); // "mundo!"
```

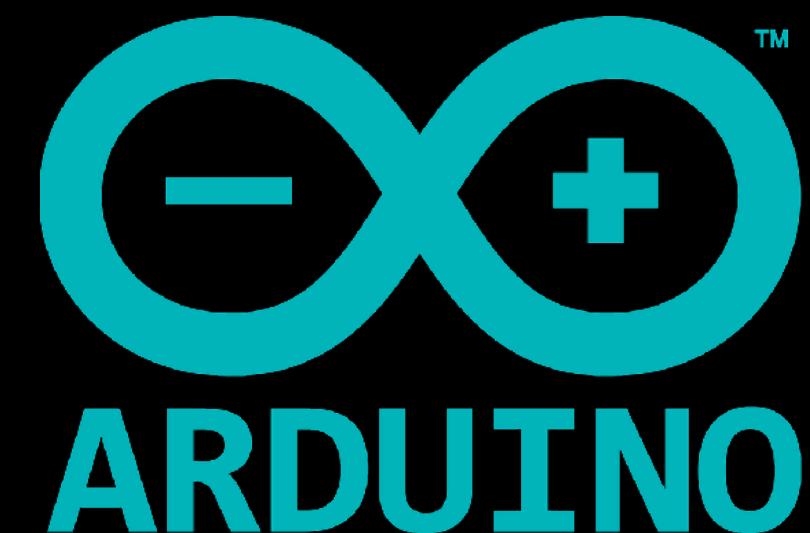
```
String texto4 = " abc abc \n";  
texto4.replace("ab", "AB"); // "ABC ABC"
```

LED

```
void setup () {  
    pinMode(pinoLED, OUTPUT);  
    digitalWrite(pinoLED, HIGH);  
}  
  
digitalWrite(pinoLED, LOW);
```

Serial

```
void setup () {  
    Serial.begin(115200); while(!Serial);  
}  
  
void loop () {  
    if (Serial.available() > 0) {  
        String texto = Serial.readStringUntil('\n');  
        Serial.println(texto);  
    }  
}
```





WiFi

```
#include <WiFi.h>

void reconnectarWiFi() {
  if (WiFi.status() != WL_CONNECTED) {
    WiFi.begin("NOME DA REDE", "SENHA DA REDE");

    Serial.print("Conectando ao WiFi...");
    while (WiFi.status() != WL_CONNECTED) {
      Serial.print(".");
      delay(1000);
    }
    Serial.print("conectado!\nEndereço IP: ");
    Serial.println(WiFi.localIP());
  }
}

void setup () {
  Serial.begin(115200); delay(500);

  reconnectarWiFi();
}

void loop () {
  reconnectarWiFi();
```



Rede WiFi

Serialização

```
JsonDocument dados;
dados["número"] = 12345;
dados["texto"] = "IoT";

String textoJson;
serializeJson(dados, textoJson);
serializeJson(dados, Serial);

String texto_json2 = "[10, 20, 30]";
JsonDocument lista;
deserializeJson(lista, texto_json2);
```

Json

```
#include <ArduinoJson.h>

JsonDocument dados;
dados["número"] = 12345;
dados["texto"] = "IoT";

String meuTexto = dados["texto"];
int meuNumero = dados["número"];

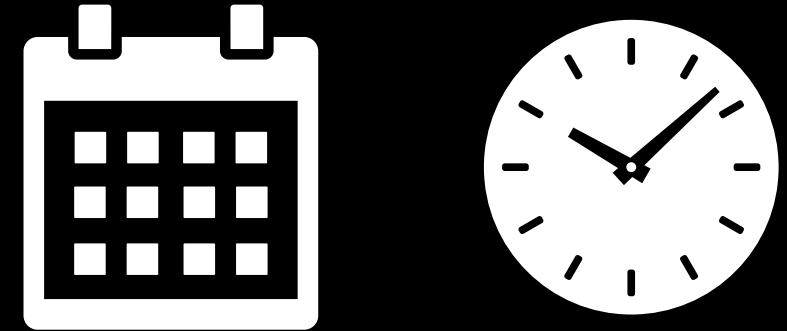
JsonDocument lista;
lista.add(10);
lista.add(20);
for (unsigned int i = 0; i < lista.size(); i++) {
  int elemento = lista[i];
  Serial.println(elemento);
}
```

O JSON

Setup

```
#include <time.h>

void setup() {
    Serial.begin(115200);
    reconnectarWiFi();
    configTzTime("<-03>3", "a.ntp.br", "pool.ntp.org");
}
```



Data / Hora (NTP)

Data/Hora Atual

```
struct tm tempo;
getLocalTime(&tempo);
```

Pedaços Individuais

```
int hora = tempo.tm_hour;
int minuto = tempo.tm_min;
int segundo = tempo.tm_sec;
int dia = tempo.tm_mday;
int mes = tempo.tm_mon + 1;
int ano = tempo.tm_year + 1900;
int diaDaSemana = tempo.tm_wday;
```

Impressão na Serial, Display, Arquivo, etc

```
Serial.println(&tempo, "%d/%m/%Y %H:%M:%S");
display.print(&tempo, "%d/%m/%Y %H:%M:%S");
arquivo.println(&tempo, "%d/%m/%Y %H:%M:%S");
```

Conversão para String

```
char buffer[100];
strftime(buffer, sizeof(buffer), "%d/%m/%Y %H:%M:%S", &tempo);
String tempoString = String(buffer);
```

Setup

```
#include <GxEPD2_BW.h>
#include <U8g2_for_Adafruit_GFX.h>
```

```
U8G2_FOR_ADAFRUIT_GFX fontes;
```

```
GxEPD2_290_T94_V2 modeloTela(10, 14, 15, 16);
```

```
GxEPD2_BW<GxEPD2_290_T94_V2, GxEPD2_290_T94_V2::HEIGHT> tela(modeloTela);
```

```
void setup() {
    tela.init();
    tela.setRotation(3);
    tela.fillRect(0, 0, 296, 128, GxEPD_WHITE);
    tela.display(true);

    fontes.begin(tela);
    fontes.setForegroundColor(GxEPD_BLACK);
}
```

Desenhos

```
tela.drawLine(x1, y1, x2, y2, cor);
```

```
tela.fillCircle(x, y, raio, cor);
```

```
tela.drawCircle(x, y, raio, cor);
```

```
tela.fillRect(x, y, comprimento, altura, cor);
```

```
tela.drawRect(x, y, comprimento, altura, cor);
```

```
tela.fillTriangle(x1, y1, x2, y2, x3, y3, cor);
```

```
tela.drawTriangle(x1, y1, x2, y2, x3, y3, cor);
```

```
tela.display(true); // SEMPRE CHAMAR NO FINAL!
```

Fontes de Símbolos

```
u8g2_font_open_iconic_all_4x_t
```



[https://github.com/olikraus/u8g2/wiki/fntpic/u8g2 font open iconic all 4x t.png](https://github.com/olikraus/u8g2/wiki/fntpic/u8g2_font_open_iconic_all_4x_t.png)

Fontes de Texto

```
u8g2_font_helvB24_te
```

```
u8g2_font_helvB18_te
```

```
u8g2_font_helvB14_te
```

```
u8g2_font_helvB12_te
```

Textos

```
fontes.setFont( u8g2_font_helvB24_te );
```

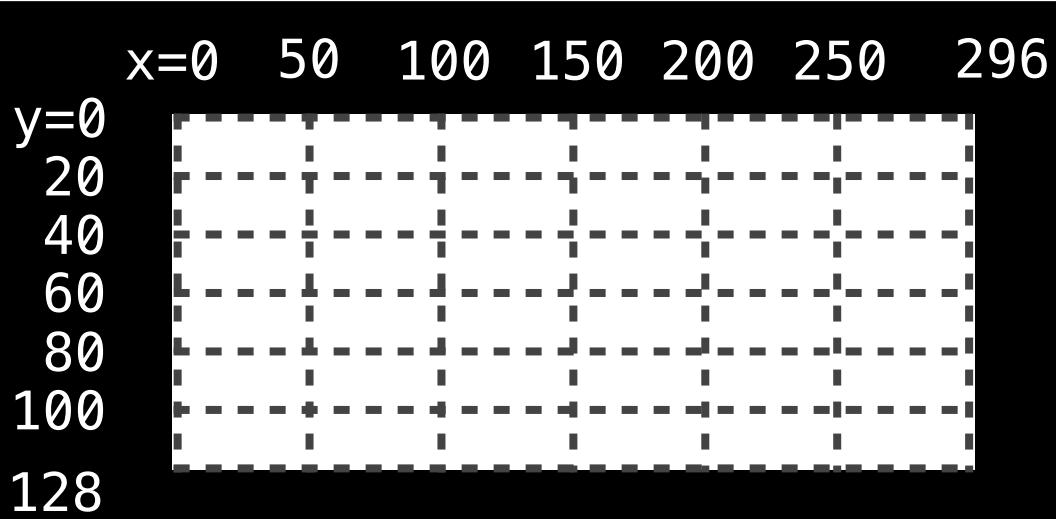
```
fontes.setFontMode(1);
```

```
fontes.setCursor(x, y);
```

```
fontes.print("Meu texto");
```

```
tela.display(true); // SEMPRE CHAMAR NO FINAL!
```

Display ePaper
2.9" WeAct



```

#include <WiFi.h>
#include <WiFiClientSecure.h>
#include "certificados.h"
#include <MQTT.h>

WiFiClientSecure conexaoSegura;
MQTTClient mqtt(1000);

void setup() {
  Serial.begin(115200);
  delay(500);

  reconnectWiFi();
  conexaoSegura.setCACert(certificado1);
}

```

```

  mqtt.begin("mqtt.janks.dev.br", 8883, conexaoSegura);
  mqtt.onMessage(recebeuMensagem);
  mqtt.setKeepAlive(10);
  mqtt.setWill("tópico da desconexão", "conteúdo");

  reconnectMQTT();
}

```

```

void loop() {
  reconnectWiFi();
  reconnectMQTT();
  mqtt.loop();
}

```

```

  mqtt.publish("tópico", "conteúdo"); // retain = false, qos = 0
  mqtt.publish("tópico2/1234/parametro", "conteúdo 2", false, 1);
}

```

Setup

Reconectar

```

void reconnectarMQTT() {
  if (!mqtt.connected()) {
    Serial.print("Conectando MQTT...");
    while(!mqtt.connected()) {
      mqtt.connect("SEU ID", "LOGIN", "SENHA");
      Serial.print(".");
      delay(1000);
    }
    Serial.println(" conectado!");

    mqtt.subscribe("topico1"); // qos = 0
    mqtt.subscribe("topico2/+parametro", 1); // qos = 1
  }
}

```



Recebimento

```

void recebeuMensagem(String topico, String conteudo) {
  Serial.println(topico + ":" + conteudo);
}

```

Envio

```

  mqtt.publish("tópico", "conteúdo"); // retain = false, qos = 0
  mqtt.publish("tópico2/1234/parametro", "conteúdo 2", false, 1);
}

```