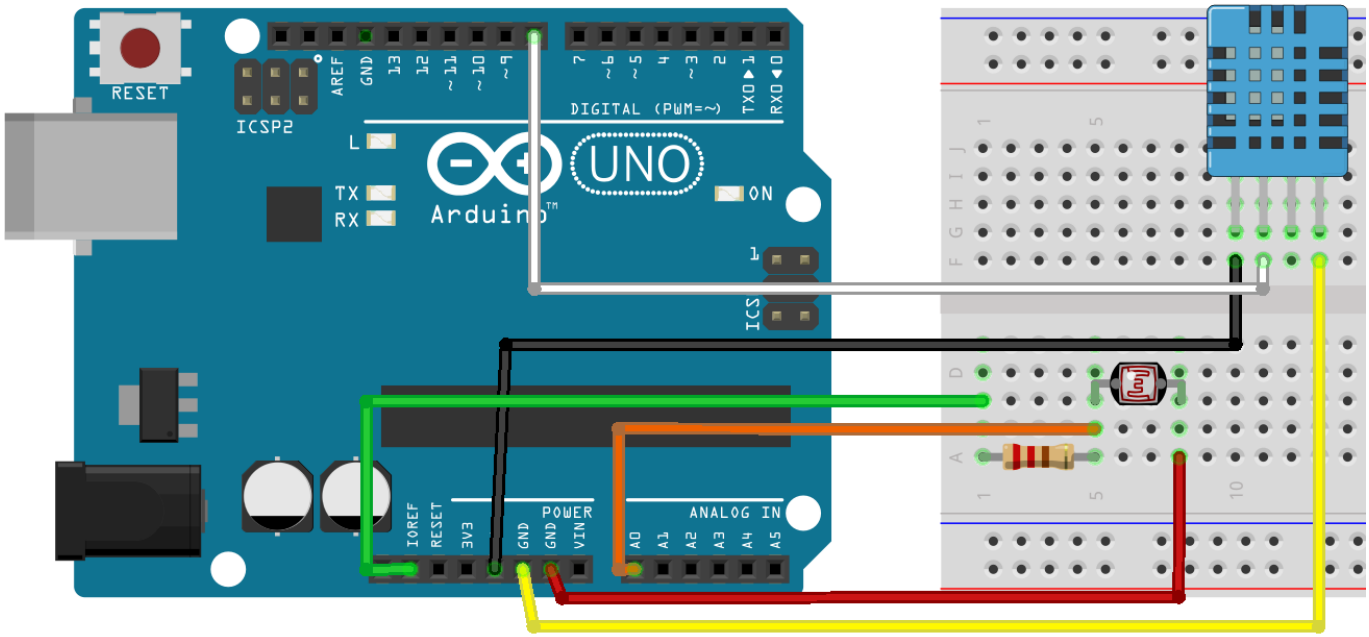
記事
Toshihiko Minamoto · 2020年9月28日 6m read

Arduino で気象観測

InterSystems ハッカソンの時、Artem Viznyuk と私のチームは Arduino ボード（1台）とその各種パーツ（大量）を所有していました。 そのため、私たちは活動方針を決めました。どの Arduino 初心者もそうであるように、気象観測所を作ることにしたのです。 ただし、Caché のデータ永続ストレージと DeepSee による視覚化を利用しました！



デバイスの操作

InterSystems の Caché は、次のようなさまざまな物理デバイスと論理デバイスを直接操作することができます。

- ターミナル
- TCP
- スプーラー
- プリンター
- 磁気テープ
- **COM ポート**
- その他多数

Arduino は通信に COM ポートを使用しているため、私たちの準備は万端でした。
一般的に、デバイスの操作は次の5つのステップに分けることができます。

1. OPEN コマンドでデバイスを現在のプロセスに登録し、デバイスにアクセスする。

2. USE コマンドでデバイスをプライマリにする。
3. 実際の操作を行う。 READ でデバイスからデータを受信し、WRITE でデータを送信する。
4. もう一度 USE でプライマリデバイスを切り替える。
5. CLOSE コマンドでデバイスを解放する。

理論上はこうなりますが、実際はどうなのでしょうか？

Caché からの点滅操作

まず、私たちはCOM ポートから数値を読み取り、指定したミリ秒の間だけ LED に電力を供給する Arduino デバイスを作りました。
回路:
C のコード（Arduino 用）

```
/* Led.ino
 * COM ??????????
 * led ? ledPin ???
 *

// led ???????
#define ledpin 8

// ???????????
String inString = "";

// ???? 1 ?????
void setup() {
    Serial.begin(9600);
    pinMode(ledpin, OUTPUT);
    digitalWrite(ledpin, LOW);
}

// ??????
void loop() {
    // com ????????
    while (Serial.available() > ) {
    int inChar = Serial.read();
    if (isDigit(inChar)) {
        // ??? 1 ??
        // ????????????
        inString += (char)inChar;
    }

    // ?????
    if (inChar == '\n') {
        // led ??????
        digitalWrite(ledpin, HIGH);
        int time = inString.toInt();
        delay(time);
        digitalWrite(ledpin, LOW);
        // ??????????
        inString = "";
    }
  }

}
```

最後に、COM ポートに 1000\n を送信する Caché のメソッドを掲載します。

```
/// 1000\n ? com ??????
ClassMethod SendSerial()
{
    set port = "COM1"
    open port:(:::" 0801n0":/BAUD=9600)     // ???????
    set old = $IO // ??????????????
    use port  // com ????????
    write $Char(10) // ?????????
    hang 1
    write 1000 _ $Char(10) // 1000\n ???
    use old // ??????????
    close port // ???????
}
```

«0801n0» は Com
ポートにアクセスするためのパラメーターを含む文字列であり、ドキュメントに記載されています。
また、/BAUD=9600 は言うまでもなく接続速度です。

このメソッドをターミナルで実行する場合、次のようになります。

```
do ##class(Arduino.Habr).SendSerial()
```

上記を実行しても何も出力されませんが、LED が 1 秒間点滅します。

データ受信

次はキーパッドを Cache に接続し、入力データを受信します。 これは、 認証委任と ZAUTHENTICATE.mac ルーチンを使用するカスタムユーザー認証として使用できます。
回路:
C のコード

```
/* Keypadtest.ino *
 * Keypad ???????????
 * Keypad ? rowPins[] ? colPins[] ???????????
 * Arduino ??????????
 *
 */

// ?????:
// https://github.com/Chris--A/Keypad
#include

const byte ROWS = 4; // 4 ?
const byte COLS = 4; // 3 ?
// ?????????????
char keys[ROWS][COLS] = {
    {'1','2','3','A'},
    {'4','5','6','B'},
    {'7','8','9','C'},
    {'*','0','#','D'}
```

```
};
// ???????? 1-8????? Arduino ??? 11-4 ??????: 1->11, 2->10, ... , 8->4
// ????? ROW0, ROW1, ROW2, ROW3 ??? Arduino ????????
byte rowPins[ROWS] = { 7, 6, 5, 4 };

// ????? COL0, COL1, COL2 ??? Arduino ????????
byte colPins[COLS] = { 8, 9, 10, 11 };

// Keypad ????
Keypad kpd = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS );

void setup() {
    Serial.begin(9600);
}

void loop() {
    char key = kpd.getKey(); // ?????????
    if(key)
    {
        switch (key)
    {
        case '#':
            Serial.println();
        default:
            Serial.print(key);
    }
    }
}
```

また、以下は同時に 1 行ずつ COM ポートからデータを取得するために使用される Caché メソッドです。

```
/// ??????????? COM1 ?? 1 ???????
ClassMethod ReceiveOneLine() As %String
{
    port = "COM1"
    set str=""
    try {
        open port:(:::" 0801n0":/BAUD=9600)
        set old = $io
        use port
        read str // ???????????????
        use old
        close port
    } catch ex {
        close port
    }
    return str
}
```

ターミナルで以下を実行します。

```
write ##class(Arduino.Habr).ReceiveOneLine()
```

また、（行末文字として送信される）#
が押されるまで入力待ち状態になります。その後、入力されたデータがターミナルに表示されます。

以上が Arduino-Caché I/O の基本です。これで、自前の気象観測所を作る準備が整いました。

気象観測所

これで気象観測所に着手できる状態になりました。 私たちはフォトレジスタと DHT11
温湿度センサを使用し、データを収集しました。

回路:

C のコード

```
/* Meteo.ino *
 * ??????????????
 * ???? COM ?????????
 * ???: H=1.0;T=1.0;LL=1;
 */

// ???????????????
int lightPin = ;

// DHT-11 ?????????
int DHpin = 8;

// DHT-11 ?????????????
byte dat[5];

void setup() {
    Serial.begin(9600);
    pinMode(DHpin,OUTPUT);
}

void loop() {
    delay(1000); // 1 ??????????
    int lightLevel = analogRead(lightPin); //????????

    temp_hum(); // ?????? dat ?????
    // And output the result
    Serial.print("H=");
    Serial.print(dat[], DEC);
    Serial.print('.');
    Serial.print(dat[1],DEC);
    Serial.print(";T=");
    Serial.print(dat[2], DEC);
    Serial.print('.');
    Serial.print(dat[3],DEC);
    Serial.print(";LL=");
    Serial.print(lightLevel);
    Serial.println(";");
}

// DHT-11 ????? dat ???
void temp_hum() {
```

```
    digitalWrite(DHpin,LOW);
    delay(30);
    digitalWrite(DHpin,HIGH);
    delayMicroseconds(40);
    pinMode(DHpin,INPUT);
    while(digitalRead(DHpin) == HIGH);
    delayMicroseconds(80);
    if(digitalRead(DHpin) == LOW);
    delayMicroseconds(80);
    for(int i=;i<4;i++)
    {
      dat[i] = read_data();
    }
    pinMode(DHpin,OUTPUT);
    digitalWrite(DHpin,HIGH);
}

// DHT-11 ??????????
byte read_data() {
    byte data;
    for(int i=; i<8; i++)
    {
        if(digitalRead(DHpin) == LOW)
        {
            while(digitalRead(DHpin) == LOW);
            delayMicroseconds(30);
            if(digitalRead(DHpin) == HIGH)
            {
                data |= (1<<(7-i));
            }
            while(digitalRead(DHpin) == HIGH);
        }
    }
    return data;
}
```

このコードを Arduino に読み込んだ後、次の形式で COM ポートからデータの送信を開始しました。

```
H=34.0;T=24.0;LL=605;
```

説明:

- H — 湿度（0〜100%）
- T — 気温（摂氏）
- LL — 輝度（0〜1023）

新しい Arduino.Info クラスを作成するため、このデータを Caché に保存します。

```
Class Arduino.Info Extends %Persistent
{

Parameter SerialPort As %String = "com1";
```

```
Property DateTime As %DateTime;

Property Temperature As %Double;

Property Humidity As %Double(MAXVAL = 100, MINVAL = 0);

Property Brightness As %Double(MAXVAL = 100, MINVAL = 0);

Property Volume As %Double(MAXVAL = 100, MINVAL = 0);

ClassMethod AddNew(Temperature = 0, Humidity = 0, Brightness = 0, Volume = 0)
{
    set obj = ..%New()
    set obj.DateTime=$ZDT($H,3,1)
    set obj.Temperature=Temperature
    set obj.Humidity=Humidity
    set obj.Brightness=Brightness/1023*100
    set obj.Volume=Volume
    write $SYSTEM.Status.DisplayError(obj.%Save())
}
```

その後は Arduino からデータを受信し、次のように Arduino.Info
クラスオブジェクトに変換するメソッドを作成しました。

```
/// ????????????????: H=34.0;T=24.0;LL=605;\n
/// Arduino.Info ????????????
ClassMethod ReceiveSerial(port = {..#SerialPort})
{
    try {
        open port:(:::" 0801n0":/BAUD=9600)
        set old = $IO
        use port
        for {
            read x //1 ?????
            if (x '= "") {
                   set Humidity = $Piece($Piece(x,";",1),"=",2)
                set Temperature =  $Piece($Piece(x,";",2),"=",2)
                set Brightness =  $Piece($Piece(x,";",3),"=",2)
                do ..AddNew(Temperature,Humidity,Brightness) // ??????
            }
        }
    } catch anyError {
        close port
    }
}
```

最後に Arduino を接続し、ReceiveSerial メソッドを実行しました。

```
write ##class(Arduino.Info).ReceiveSerial()
```

このメソッドは、Arduino から無期限にデータを受信して保存します。

データの視覚化

デバイスを作成後、室外に設置して夜間にデータを収集するようにしました。

翌朝には 36000 件以上のレコードを取得できましたので、そのデータを DeepSee で MDX2JSON
サーバーサイド REST API と DeepSeeWeb
ダッシュボードレンダラーを使用して視覚化することにしました。結果は次のとおりです。

以下は輝度です。 5:50 頃に日の出をはっきりと確認できます。

以下は気温と湿度のグラフです。

湿度と気温の負の相関がはっきりとわかります。

デモ

こちらからアクセスできます。

まとめ

InterSystems Cachéを使用すると、多数の異なるデバイスと直接通信できます。
データ処理および視覚化ソリューションを迅速に開発できます。自前の気象観測所を作成して Caché
に接続し、結果を視覚化するまでには約 4 時間かかりましたが、その大部分は回路の設計と C
のコーディングに費やした時間です。

リンク

» ドキュメント
» GitHub リポジトリ

#ターミナル #Caché

ソースURL:
https://jp.community.intersystems.com/post/arduino-%E3%81%A7%E6%B0%97%E8%B1%A1%E8%A6%B3%E6%B8%AC