arduino-peripherals Documentation

Release latest

May 28, 2019

Contents:

This library provides a number of interfaces to simple input and output devices like buttons, LEDs, buzzers, thermistors, etc.

Please see ReadTheDocs for the latest documentation.

CHAPTER 1

Introduction

CHAPTER 2

Device installation

In this section we cover retrieval of the latest release or development version of the code and subsequent installation for an Arduino device.

2.1 Download

2.1.1 Latest release

Navigate to the latest release and either download the `.zip` or the `.tar.gz` file.

Unpack the downloaded archive.

2.1.2 From source

The source is hosted on GitHub, to install the latest development version, use the following command.

```
git clone https://github.com/jfjlaros/arduino-peripherals.git
```

2.2 Installation

2.2.1 Arduino IDE

In the Arduino IDE, a library can be added to the list of standard libraries by clicking through the following menu options.

- Sketch
- Import Library
- Add Library

To add the library, navigate to the downloaded folder and select the subfolder named `peripherals`.

- Click OK.

Now the library can be added to any new project by clicking through the following menu options.

- Sketch
- Import Library
- peripherals

2.2.2 Ino

Ino is an easy way of working with Arduino hardware from the command line. Adding libraries is also easy, simply place the library in the `lib` subdirectory.

```
cd lib
git clone https://github.com/jfjlaros/arduino-peripherals.git
```

CHAPTER 3

Usage

3.1 Generic input and output

Most of the libraries described here inherit from either a generic input or output class.

3.1.1 Input

The constructor for input devices (e.g., a button or photoresistor) takes up to three parameters of which two are optional.

Table 1: Generic input constructor parameters.

parameter	description	mandatory	default
0	Pin number.	yes	
1	Invert input behaviour.	no	`false`
2	Use internal pullup resistor.	no	`false`

Every input device has at least the following functions.

Table 2: Generic input functions.

name	description
`analogRead`	Read an analogue value.
`digitalRead`	Read a digital value.
`on`	Check whether the state is HIGH.
`off`	Check whether the state is LOW.

3.1.2 Output

The constructor for output devices (e.g., a buzzer or an LED) takes up to two parameters of which one is optional.

Table 3: Generic output constructor parameters.

parameter	description	mandatory	default
0	Pin number.	yes	
1	Invert output behaviour.	no	`false`

Every output device has at least the following functions.

Table 4: Generic output functions.

name	description
`analogWrite`	Write an analogue value.
`digitalWrite`	Write a digital value.
`on`	Write HIGH.
`off`	Write LOW.

3.2 Button

Include the header file to use the button library.

```
#include <button.h>
```

3.2.1 Example

We make a `Button` instance that uses pin `10` follows.

```
Button button(10);
```

Check whether the button is pressed by using the `on()` function.

```
if (button.on()) {
  // Button is pressed.
}
```

3.3 Buzzer

Include the header file to use the buzzer library.

```
#include <buzzer.h>
```

3.3.1 Example

We make a `Buzzer` instance that uses pin `10` as follows.

```
Buzzer buzzer(10);
```

To generate a tone of 10,000Hz, use the `tone()` member function.

```
buzzer.tone(10000);
```

The buzzer can be turned off with the `noTone()` member function.

```
buzzer.noTone();
```

3.4 LED

Include the header file to use the LED library.

```
#include <led.h>
```

3.4.1 Example

We make a `LED` instance that uses pin `10` as follows.

```
LED led(10);
```

The led can be turned on or off.

```
led.on();
led.off();
```

If the pin supports analogue output, the LED can also be set to a specific brightness.

```
led.analogWrite(20);
```

3.5 PhotoResistor

Include the header file to use the photoresistor library.

```
#include <photoresistor.h>
```

3.5.1 Example

We make a `PhotoResistor` instance that uses pin `10` as follows.

```
PhotoResistor photoresistor(10);
```

The value of the photoresistor can be read with the `analogRead()` function.

```
photoresistor.analogRead();
```

3.6 Thermistor

Include the header file to use the thermistor library.

```
#include <thermistor.h>
```

3.6.1 Example

The constructor for the thermistor is slightly different from the generic one. It takes an additional mandatory parameter named `resistor` with which the resistor value (in Ω) should be passed.

We make a `Thermistor` instance that uses pin `10` and uses an 100Ω resistor as follows.

```
Thermistor thermistor(10, 100.0);
```

The temperature can be read using various functions.

```
thermistor.kelvin();
thermistor.celsius();
thermistor.fahrenheit();
```

3.7 Microphone

Include the header file to use the microphone library.

```
#include <microphone.h>
```

3.7.1 Example

We make a `Microphone` instance that uses pin `A4` as follows.

```
Microphone microphone(A4);
```

The sound level of the microphone can be read with the `soundLevel()` function which takes a number of samples and determines the range of these samples. If we want to determine the sound level based on 1024 samples, we do the following.

```
microphone.soundLevel(1024);
```

CHAPTER 4

Contributors

- Jeroen F.J. Laros <jlaros@fixedpoint.nl> (Original author, maintainer)

Find out who contributed:

```
git shortlog -s -e
```