DualEncoders

TUTORIAL

WHAT IS DualEncoders?

The DualEncoders Library allows you to use dual concentric rotary encoders with 2 steps per detent, like those sold by Propwash.

HOW DOES IT WORK?

First we wire the switches in a specific manner (which will be explained later). Then, we tell the DualEncoders Library which Arduino pins we will use :

```
DualEncoders enc(4,5,3,2,6);
```

The encoders are designated as *inner* or *outer*. The Library's methods will return in which direction each encoder was turned : *CW* or *CCW*, or 0 (zero) if it has not turned or reached a detent (click).

```
direction = enc.inner();
```

HOW DO WE WIRE THE ENCODERS?

Each encoder has 3 pins : A, B and com. Pins A and B are connected to a digital pin on the Arduino, and the common (com) pin is connected to ground. One of the switch pin is connected to a digital pin on the Arduino and the other one is connected to ground. (It does not matter which one is connected to what.)

```
//=======================================================================================
// Encoder   Arduino
// outerB-------2
// outerA-------3
//    Com--+------+--Gnd
//    Com--+      |
// innerA-------4 |
// innerB-------5 |
// Switch-------6 |
// Switch---------+
//----------------------------------------------------------------------------------------
```

WHAT ARE THE METHODS THAT THE MENU LIBRARY PROVIDES?

MANDATORY METHOD

DualEncoder::DualEncoder(int innerA, int innerB, int outerA, int outerB, int switch) (MANDATORY)

```
DualEncoders enc(4,5,3,2,6);
```

The first method is called a constructor, which creates the encoders. It has to be placed **before** setup().
innerA, innerB, outerA and outerB, corresponds to the pins which you used in your setup;
switch corresponds to the pin you used in your setup.

You can have as many dual encoders as you wish. Just give them different names.

METHODS TO READ THE ENCODERS

int DualEncoders::readInner()

```
innerEncoder = enc.readInner();
```

You will use this method to get the direction of rotation of the inner encoder. It will return *CW* or *CCW*, or 0 (zero) if it has not turned or reached a detent (click).

int DualEncoders::readOuter()

```
innerEncoder = enc.readOuter();
```

You will use this method to get the direction of rotation of the outer encoder. It will return *CW* or *CCW*, or 0 (zero) if it has not turned or reached a detent (click).

int DualEncoders::readSwitch()

```
encoderSwitch = enc.readSwitch();
```

You will use this method to find out if the encoder’s switch is actualy pressed. If it is, it will wait for you to release the switch, and return zero. If it is not, it will return 1.

EXAMPLE SKETCH

```
#include <DualEncoder.h>

DualEncoder com1(4,5,3,2,6);

//setup=====================================================
void setup() {
  Serial.begin(9600);          //To display on the monitor
  Serial.println("Starting");
}//setup----------------------------------------------------

//loop=========================================================================
void loop() {
  int stat = com1.readInner();
  if (stat == CCW) Serial.println("sim/radios/stby_com1_coarse_down");
  if (stat ==  CW) Serial.println("sim/radios/stby_com1_coarse_up");
  stat = com1.readOuter();
  if (stat == CCW) Serial.println("sim/radios/stby_com1_fine_down");
  if (stat ==  CW) Serial.println("sim/radios/stby_com1_fine_up");
  if (com1.readSwitch() == 0) Serial.println("sim/radios/com1_standy_flip");
}//loop-----------------------------------------------------------------------
```

I sincerely hope that this DualEncoder Library will help you in your projects.
Jacques Bellavance