MTD2A: Mobil Train Detection and Action – arduino libray

MTD2A: Model **T**rain **D**etection **A**nd **A**ction – arduino library https://github.com/MTD2A/MTD2A
Jørgen Bo Madsen / V1.2 / 31-05-2025

MTD2A er en samling af avancerede og funktionelle C++ klasser - byggeklodser - til tidsstyret håntering af input og output. Biblioteket er tiltænkt Arduino interesserede uden større programmeringserfaring, der har elektronikstyring og -automatisering som interesse, samt modeltog som hobby.

Fælles for alle byggeklodser:

- Uunderstøtter en bred vifte af inputsensorer og outputenheder
- Er enkle at bruge til at bygge komplekse løsninger med få kommandoer
- Fungere Ikke-blokerende, procesorienteret og tilstandsdrevet
- Tilbyder omfattende kontrol- og fejlfindingsinformation
- Grundigt dokumenterede med eksempler

Bibliotek

```
#include <MTD2A.h>
using namespace MTD2A_const;
```

Nuværende byggeklodser

- MTD2A_binary_input.h
- MTD2A_binary_output.h

Yderligere planlgate byggeklodser

- MTD2A_delay.h
- MTD2A_astable.h
- MTD2A_flip_flop.h
- MTD2A_tone.h
- MTD2A_sound.h
- MTD2A_servo.h
- MTD2A_stepper.h
- MTD2A_display.h
- MTD2A_ultrasonic.h
- MTD2A_laser.h
- MTD2A_IR_ranging.h
- MTD2A_DCC_input.h

Globale variable

```
ENABLE        = true,  DISABLE      = false;
ACTIVE        = true,  COMPLETE     = false;
FIRST_TRIGGER = true,  LAST_TRIGGER = false;
TIME_DELAY    = true,  MONO_STABLE  = false;
NORMAL        = true,  INVERTED     = false;
PULSE         = true,  FIXED        = false;
BINARY        = true,  PWM          = false;
```

Proces faser

```
RESET_PHASE      = 0;
BEGIN_PHASE      = 1, OUTPUT_PHASE    = 2, END_PHASE      = 3
FIRST_TIME_PHASE = 1, LAST_TIME_PHASE = 2, BLOCKING_PHASE = 3
COMPLETE_PHASE   = 4;
```

MTD2A_base

Klassen indeholder fælles interne funktioner, globale fælles funktioner og -variable.

MTD2A::set_globalDebugPrint (); Aktiver fejl og status medddelelser til Arduiono IDE Serial Monitor.
Parameter = ({ENABLE | DISABLE}) Udelades parameter sættes default ENABLE

MTD2A::set_delayTimeMS (); Sætter hastigheden hvormed alle instantierede objekter opdtateres.
Parameter = ({ DELAY_NORMAL | DELAY_FAST}) Udelades parameter sættes default DELAY_NORMAL
Små microprocessoresom fx MEGA,UNO og NANO bør benytter DELAY_10MS og ESP32 bør benytte DELAY_1MS Hvis der der er behov for at detektere hurtige reaktioner fra fx infrarøde sensore, bør delayTimeMS sættes til DELAY_1MS

MTD2A::set_loop_execute (); Opdaterer alle instantierede objekte (Linked list of function pointers)
Absolut sidste kommando i void loop (); og skal altid kaldes en - og kun - en gang.

Eksemplel på parallel processering

```
// Two blinking LEDs. One with symmetric interval and another with asymetric interval.
// Jørgen Bo Madsen / may 2025 / https://github.com/jebmdk

#include <MTD2A.h>
using namespace MTD2A_const;

MTD2A_binary_output red_LED   ("Red LED",   400, 400);
// 0.4 sec light, 0.4 sec no light
MTD2A_binary_output green_LED ("Green LED", 300, 700, 0, PWM, 96);
// 0.3 sec light, 0.7 sec no light, PWM dimmed

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

  red_LED.initialize   (9);   // Output pin 9
  green_LED.initialize (10);  // Output pin 10

  Serial.println("Two LED blink");
}

void loop() {
  if (red_LED.get_processState()   == COMPLETE)   red_LED.activate();
  if (green_LED.get_processState() == COMPLETE) green_LED.activate();

  MTD2A::loop_execute();
} // Two blinking LEDs. One with symmetric interval and another with asymetric interval.
```