Arduino

LED

HW1 Feedback

- Požiadať o revíziu
- Červené a biele komentáre
- Môžete použiť min() a abs(), alebo implementovať sami
- positive_stars(), negative_stars()
- Pomenovať konštanty [☠2]

Pravidlo 2

- “Important constant values (pin numbers, animation periods, ...) should be declared as constants (not directly embedded as literals in code). “
- Kedy?
  - Hodnota s explicitným významom, ktorý ale nie je zrejmý z kódu
  - potenciál na zmenu, hodnota použitá aspoň raz v kóde
- Výhody
  - Self-documentation
  - Ľahšie zmeny, DRY
- Upresnenie
  - Menná konvencia UPPER_CASE
  - Môžeme použiť aj na zložitejšie objekty ako literály
  - Preferujte `constexpr` pred `const` (v C++, C23+)

```
void func(){
  for(int i = 0, i < 5, i++){
    digitalWrite(13 - i, LOW);
    }
}
```

```
void func(){
  for(int i = 0, i < LEDS_COUNT, i++){
    digitalWrite(LEDS[i], LOW);
    }
}
```

Arduino

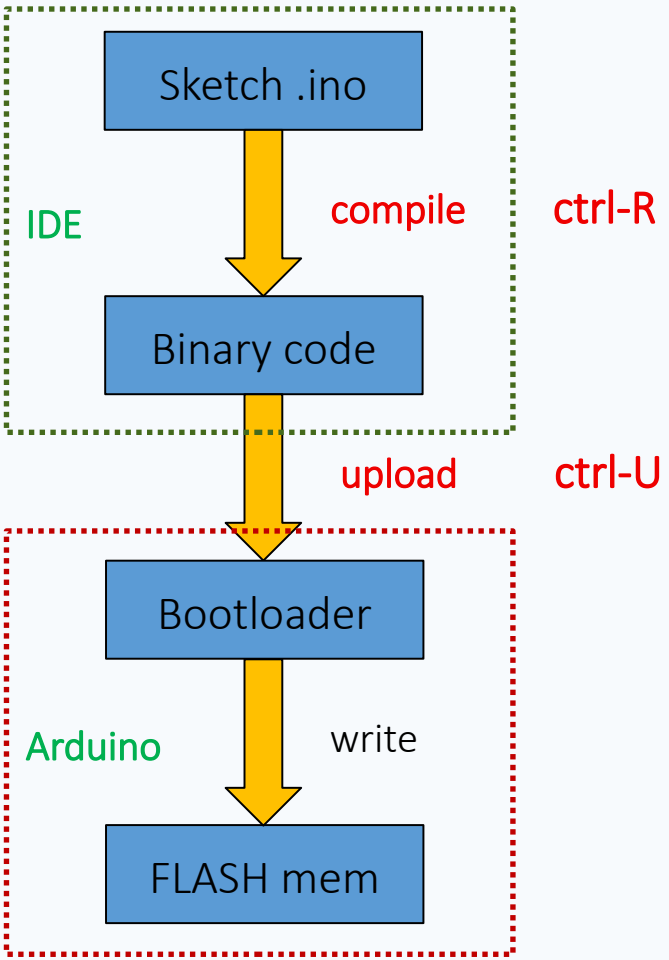
- Arduino UNO
  - základní deska, digitání a analogové piny
- expansion board (shield)
  - multifunction shield - funduino
  - tlačítka, LED, 4-místný displej, bzučák
- Arduino IDE
  - www.arduino.cc
  - kabinet.fyzika.net/dilna/ARDUINO/funduino-popis.php
  - 2.3.x
  - list box - Arduino Uno / port
    - automaticky
    - nebo select another / Uno / Serial port USB
  - (manual setup)
    - manage libraries
    - Arduino AVR Boards (Uno)
    - Tools / Board / ... Arduino Uno
    - Tools / Port / COMx (Arduino Uno)

1.000.000x míň, než PC

```
CPU ATmega328P
14 digital I/O pins
6 analog inputs
Clock speed 16 MHz
FLASH memory 32 KB
SRAM 2 KB
EEPROM 1 KB
```

Arduino

- sketch
  - save -> Documents/Arduino/***novak*** -> ***novak***.ino
  - compile ctrl-R
  - upload ctrl-U
- void **setup**();
  - jednou při startu
  - nastavení módu PINů
  - počáteční hodnoty PINů / proměnných
- void **loop**();
  - ~ 1000x /s
  - vlastní výkonný kód
    - = volání funkcí




```
int main() {
  init();
  setup();
  for(;;)
    loop();
}
```

```
void setup() {
  ....
}

void loop() {
  ....
}
```

inicializace provede se jednou

smyčka provádí se cca 1000x /s

zjednodušeno

nezastavovat, co nejrychleji se vrátit

LED

- diody připojeny na piny
- inicializace pinu
  - typicky v setup()
  - všechny piny inicializovat
  - **pinMode**(pin, **OUTPUT**)
- ovládání LED ≈ zápis na PIN
  - **digitalWrite**(pin, HIGH/LOW)
  - HIGH ≈ ☻ nesvítí
  - LOW ≈ ☻ svítí
- ➡ rozsviťte i-tou LED
  - ostatní zhasněte
  - parametr

číslo pinu, ne 0, 1, ... !

```
// my funshield library

constexpr int led[] {..};
constexpr int led_count = ..;

// my assignments

void led_on(..) {
  ....
}

// setup and loop

void setup() {
  ....
}

void loop() {
  ....
}
```

výkonný kód

pinMode

inicializace ≡ zhasnutí

no copy-and-paste

čísla nepoužívat nikde jinde! [☠2]

```
// my funshield library
constexpr int led[] { 13, 12, 11, 10 };
constexpr int led_count = sizeof(led)/sizeof(led[0]);

// ....
  digitalWrite( led[i], LOW);
```

[☠1] jen volání vlastních funkcí

low-level kód [☠1]
nepoužívejte přímo, vytvořte si abstrakci

Blikání a časování

- unsigned long **millis**();
  - počet ms od staru

- ➟ 2.1 blikající LED
  - ? kdy něco dělat
  - ? co dělat
    - (interní) stav vs. jeho vizualizace
  - ? co si potřebuju pamatovat
  - ? kde to budu mít uložené

- ➟ 2.2 semafor na železničním přejezdu
  - vždy jedna dvojice LED svítí, druhá ne
  - parametr: čas jednoho bliku

```
unsigned long last_time = 0;

bool timer( unsigned long interval) {
  auto now = millis();
  if( now >= last_time + interval) {
    last_time += interval;
    ....
  }
  return ....
}
```

globální musí přežít!

nastal čas něco dělat!

teď změna stavu

0	300	600	900
☺	☺	☺	☺

vizualizace

```
void blik(....) {
  bool on = false;
  if( is_time) {
    on = ! on;
    // vizualizace
  }
}
```

funguje?

```
bool on = false;

void blik(....) {
  if( is_time) {
    on = ! on;
    ....
  }
}
```

stav musí přežít

změna stavu

dekom pozice [☠1]

logická negace

Elegance a kvalita kódu

co je na tomto kódu špatně?

```
if( condition) {
  digitalWrite( led[i], LOW);
} else {
  digitalWrite( led[i], HIGH);
}
```

nic (?)
zkompiluje se bez warningů
dělá to, co má, efektivně

neelegantní kopírování [☠3]
hůře udržovatelný
nízká úrovní abstrakce
opakující se fragmenty

když platí nějaká podmínka, tak zapíšu na pin, který mám uložený v nějakém poli, nízkou hodnotu; když neplatí, tak vysokou ☹

```
ledOnOff( i, podminka);
```

rozsviť/zhasni i-tou diodu [☠1]
(podle nějaké podmínky) ☺

~~na jakém pinu jaká dioda~~
~~v jakém poli~~
~~jaké hodnoty pro rozsvícení~~
změna hw- jiné piny, jiný počet diod
kód snadno čitelný
funkční dekompozice

ctrl-c ctrl-v

[☠3]

Elegance a kvalita kódu

co je na tomto kódu špatně?

```
void ledOnOff( int led_index, bool cond) {
  if( condition) {
    digitalWrite( led[i], LOW);
  } else {
    digitalWrite( led[i], HIGH);
  }
}
```

interní technické detaily schované ☺

kód stále neelegantní kopírování ☹

```
digitalWrite( led[i], cond ? LOW : HIGH);
```

podmínka uvnitř složitějšího výrazu

?:
ternární operátor
výrazový if

```
constexpr int ledValue[] { HIGH, LOW };

void ledOnOff( int led_index, bool cond) {
  digitalWrite( led[i], ledValue[cond]);
}
```

alternativa:
podmínka 〰 indexace
vhodné při více hodnotách

Elegance a kvalita kódu

```
void kulicky( unsigned long mydelay) {
  auto now = millis();
  if( now >= last_time + mydelay) {
    last_time += mydelay;
  ....

void snake( int len, unsigned long mydelay) {
  auto now = millis();
  if( now >= last_time + mydelay) {
    last_time += mydelay;
  ....

void ....
  auto ....
```

ctrl-c ctrl-v

[☠3]

```
void kulicky( unsigned long mydelay) {
  if( timer( mydelay)) {
    ....
```

```
for( int i = 0; i < 4; ++i) {
  ledOnOff( i, ...);
```

Co je to '4'? Čeho 4?
Je to stejná '4' jako
ta o dva řádky výš? [☠2]

V kódu by neměly být konstanty přímo hodnotou (snad kromě 0 a 1). Všechny konstanty by se měly srozumitelně jmenovat.

Kulička a had

- 2.4 běhající kulička
  - 1. parametr:
    - čas zobrazení jedné kuličky v ms
  - 2. parametr: způsob běhání
    - dokola: 01230123...
    - odráží se: 012321012321...
  - obecné (parametrizované) řešení
    - **nevázané** na pevný počet kuliček

- ➡ 2.5 ❤ běhající had
  - 2.5a několik kuliček (LED) za sebou
    - parametr: velikost hada = počet LED
    - kulička z 2.4 ≈ had velikosti 1
    - had postupně vylézá a zalézá
      - nejdřív se zobrazí jedna kulička, pak dvě, ...
  - 2.5b aliasing
    - krajní kuličky část času s nižší intenzitou
    - svítí menší počet cyklů 〰 nižší intenzita
    - pokročilejší: rostoucí / klesající intenzita
  - 2.5c PWM ➡ www.arduino.cc

- ➡ kvalita kódu
  - parametrizace
  - konzistence, dekompozice [☠0,1]
  - konstanty [☠2]

i příklady ze cvičení

```
// my funshield library
constexpr int led[] {..};
bool timer( .. interval) {}

// my assignments
....
void x23a_binary_led( int n) {}
void x23b_count_up( int delay) {}
void x24_dot() {}
void x25_snake( int len) {}

// setup and loop
void setup() { .... }
void loop() {
  ....
  // x23b_count_up( 50);
  x24_dot( 300, true);
  // x25_snake( 2);
}
```