

# Arduino a spol.

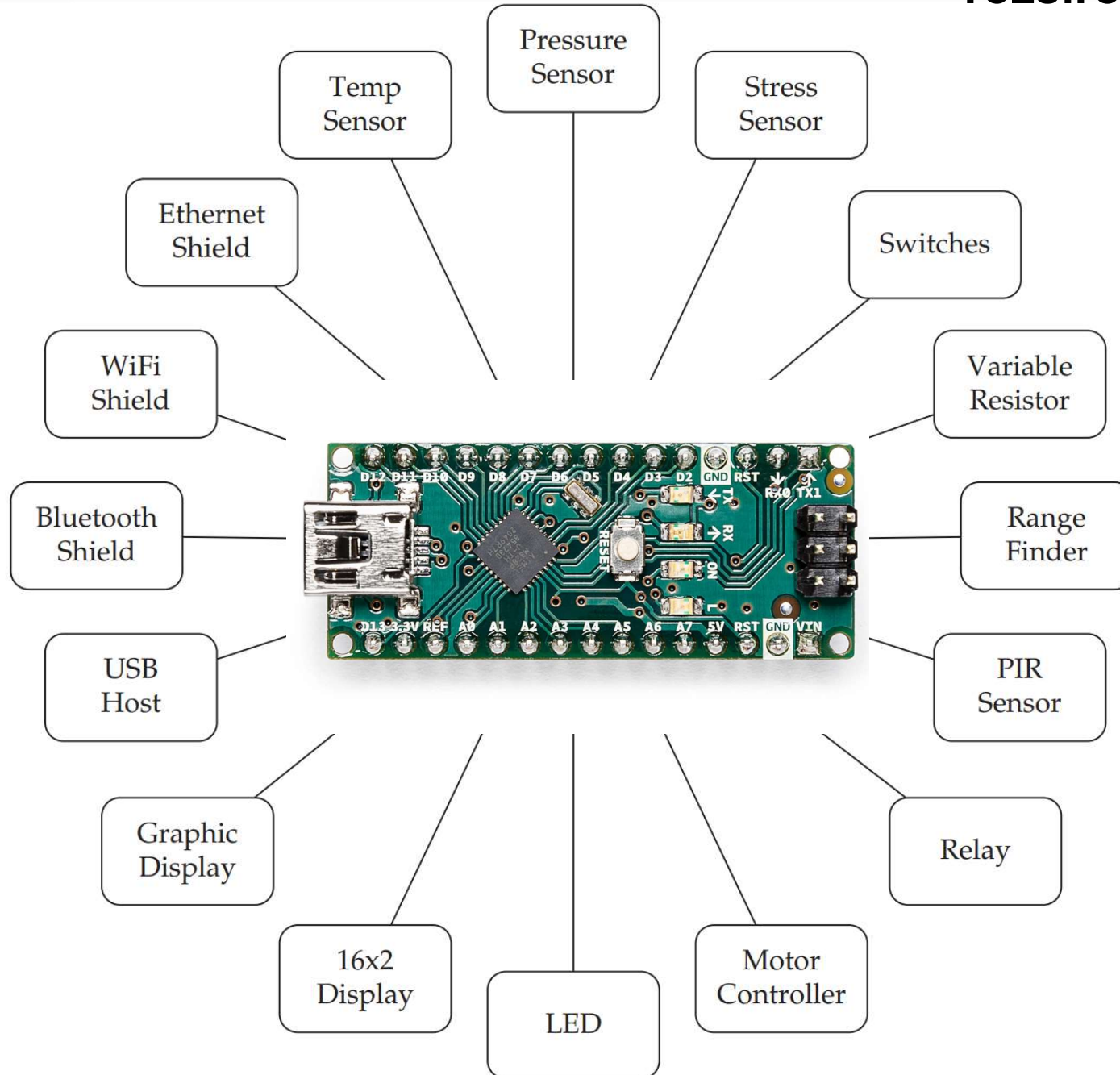
- **Arduino a jeho odvozené varianty, hw & sw**
- **instalace integrovaného vývojového prostředí (IDE)**
- **demonstrační modul Seeduino Xiao**
- **první programy – Blink a Hello World**
- **základy programování**
- **použití knihoven pro začlenění dalších komponent**
- **tlačítka a digitální rozhraní**

# Proč Arduino?

- standardní mikrokontroler pro hobby komunitu, výuku i vědecké projekty – snadné použití, nízká cena a velké množství doplňků „**shields**“ – rozšíření funkcí
- „open source“ koncepce – kdokoliv může vyrábět a prodávat odvozené moduly – klony
- jednoduché programovací prostředí IDE
  - programování na bázi Processing – vlastní jazyk se nazýval **Wiring**, jde vlastně o C či C++ (pokud je dost místa)
  - lze přidat doplňkové softwarové moduly i pro další oblíbené „kompatibilní“ mikrokontrolerové systémy
  - rozsáhlé množství knihoven pro specifické periferie (sensory, motory, LED pásy) – snadné začlenění do projektu
- obrovská uživatelská komunita – dostupné informace

# Interfacing Arduino

- široká škála možností rozšiřování

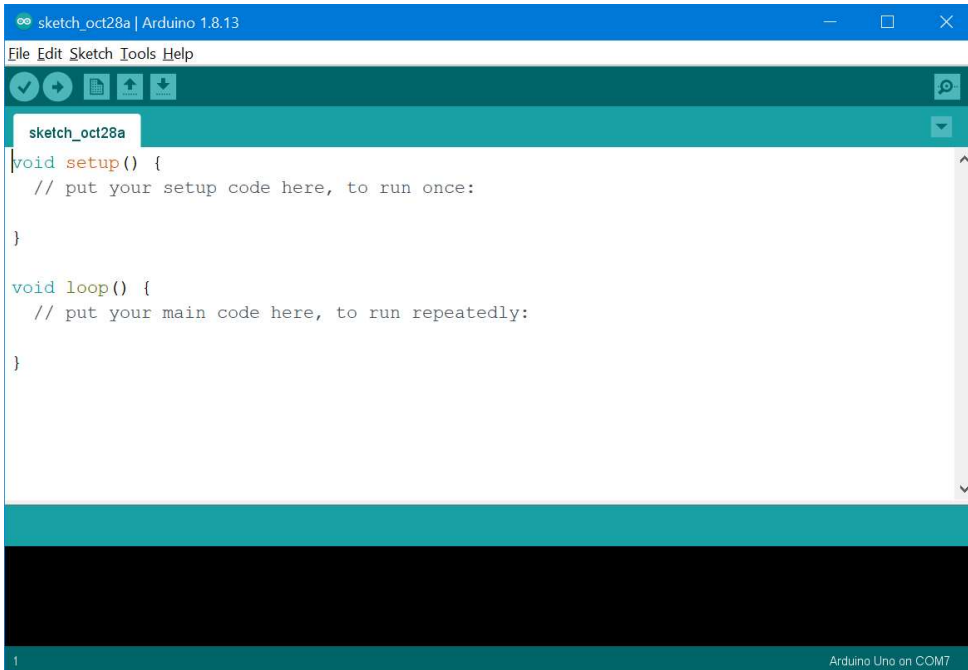


# Instalace IDE

- relevantní detailní informace jsou dostupné na webu:

<https://www.arduino.cc/en/Guide>

- instalační verze jsou dostupné pro Windows, Linux i Mac
- pokud se instalace podařila, můžeme program otevřít:



The screenshot shows the Arduino IDE window titled "sketch\_oct28a | Arduino 1.8.13". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". Below the menu bar is a toolbar with icons for opening, saving, and running. The main editor area contains the following code:

```
sketch_oct28a
void setup() {
  // put your setup code here, to run once:
}

void loop() {
  // put your main code here, to run repeatedly:
}
```

At the bottom of the window, there is a dark terminal area with the text "Arduino Uno on COM7" visible.

*nahoře je menu a ovládací tlačítka*

*v prostřední textové části se píše vlastní program*

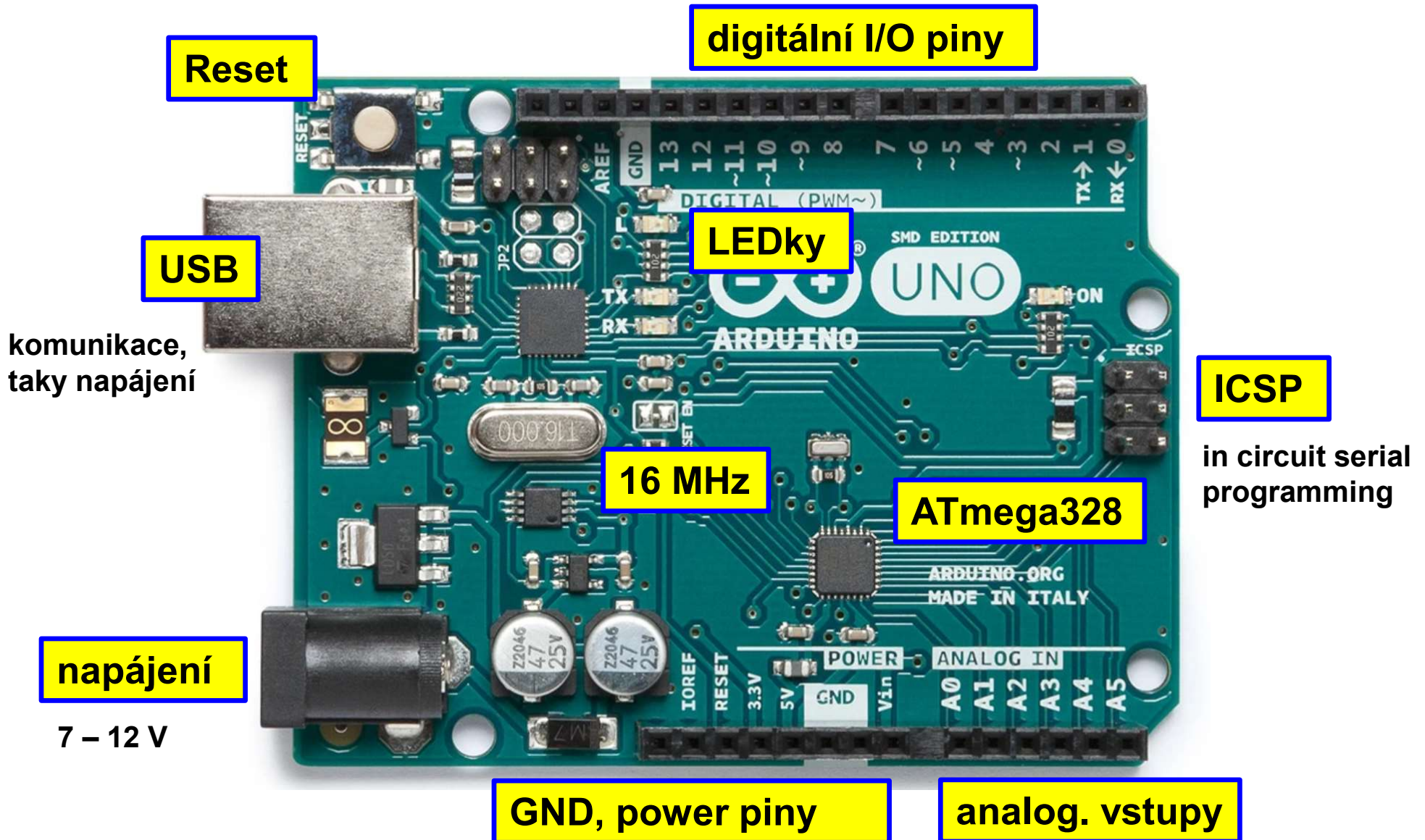
*v dolní části se objevují systémové informace z kompilace a zavádění kódu do mikrokontroleru, a chyby*

- při spolehlivém (a rychlém) připojení k internetu je možnost použít alternativní online IDE (Arduino Web Editor)

# Které Arduino?

[store.arduino.cc/collections/boards](https://store.arduino.cc/collections/boards)

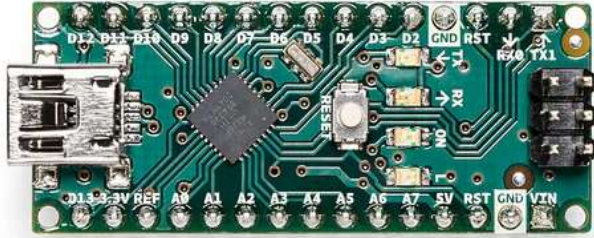
- aktuálně lze za klasiku považovat Arduino UNO (rev. 3, SMD):



[store.arduino.cc/collections/boards/products/arduino-uno-rev3-smd](https://store.arduino.cc/collections/boards/products/arduino-uno-rev3-smd)



# malé ...



## Arduino Nano

~ 3 USD z Aliexpress

18\$

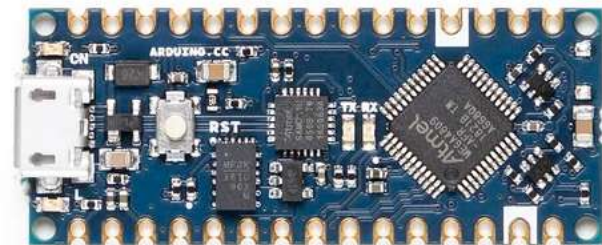
The Arduino Nano is a small, complete, and breadboard-friendly board based on the ATmega328 (Arduino Nano 3.x). It has more or less the same functionality of the Arduino Duemilanove, but in a different package. It lacks only a DC power jack, and works with a Mini-B USB cable instead of a standard...



## Arduino Micro

18\$

The Micro is a microcontroller board based on the ATmega32U4 (datasheet), developed in conjunction with Adafruit. It has 20 digital input/output pins (of which 7 can be used as PWM outputs and 12 as analog inputs), a 16 MHz crystal oscillator, a micro USB connection, an ICSP header, and a reset button...



## Arduino Nano Every

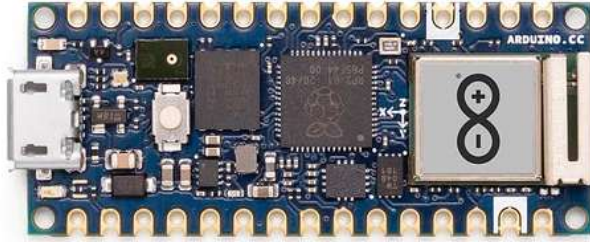
11\$

The Nano Every is Arduino's 5V compatible board in the smallest available form factor: 45x18mm! The Arduino Nano is the preferred board for many projects requiring a small and easy to use microcontroller board. The small footprint and low price, make the Nano Every particularly suited for wearab...

- základní jednoduché varianty, pracují na 5 V (klasická TTL úroveň)



# mocné ...



## Arduino Nano RP2040 Connect

21\$

Meet the only connected RP2040 board. It fits the Arduino Nano form factor, making it a small board with BIG features. The brain of the board is the Raspberry Pi RP2040 silicon; a dual-core Arm Cortex M0+ running at 133MHz. It has 264KB of SRAM, and the 16MB of flash memory is off-chip



## Arduino Due

35\$

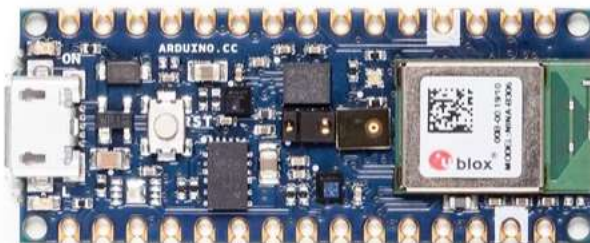
The Arduino Due is a microcontroller board based on the Atmel SAM3X8E ARM Cortex-M3 CPU. It is the first Arduino board based on a 32-bit ARM core microcontroller. It has 54 digital input/output pins (of which 12 can be used as PWM outputs), 12 analog inputs, 4 UARTs (hardware serial ports), a 84 ...



## Arduino MKR WiFi 1010

28\$

The Arduino MKR WiFi 1010 is the easiest point of entry to basic IoT and pico-network application design. Whether you are looking at building a sensor network connected to your office or home router, or if you want to create a BLE device sending data to a cellphone



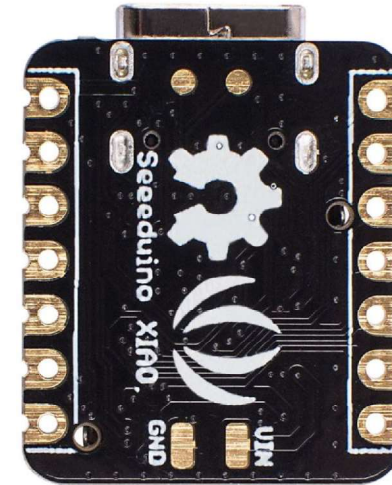
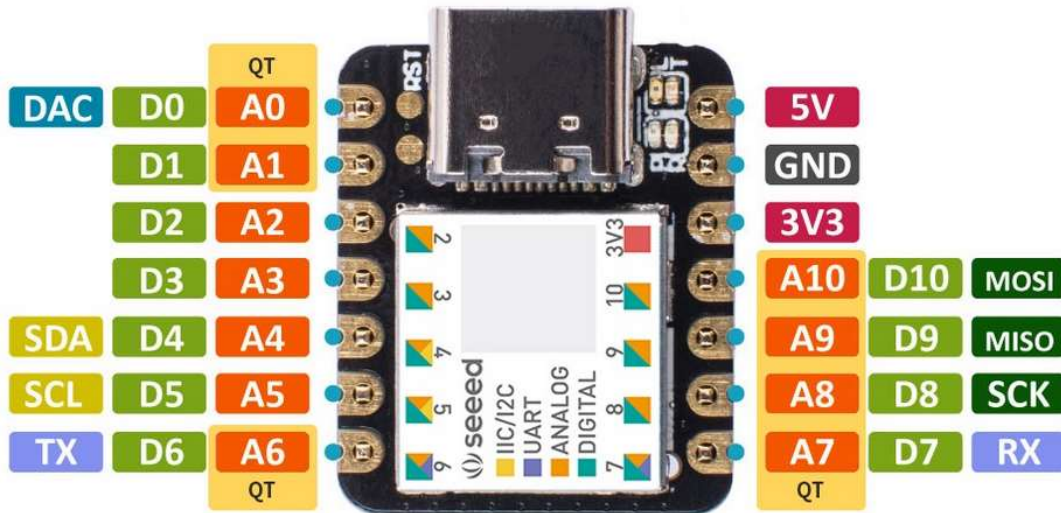
## Arduino Nano 33 BLE Sense

27\$

The Nano 33 BLE Sense (without headers) is Arduino's 3.3V AI enabled board in the smallest available form factor: 45x18mm! The Arduino Nano 33 BLE Sense is a completely new board on a well-known form factor. It comes with a series of embedded sensors: 9 axis inertial sensor

# Seeeduino Xiao

[wiki.seeedstudio.com/Seeeduino-XIAO/](http://wiki.seeedstudio.com/Seeeduino-XIAO/)

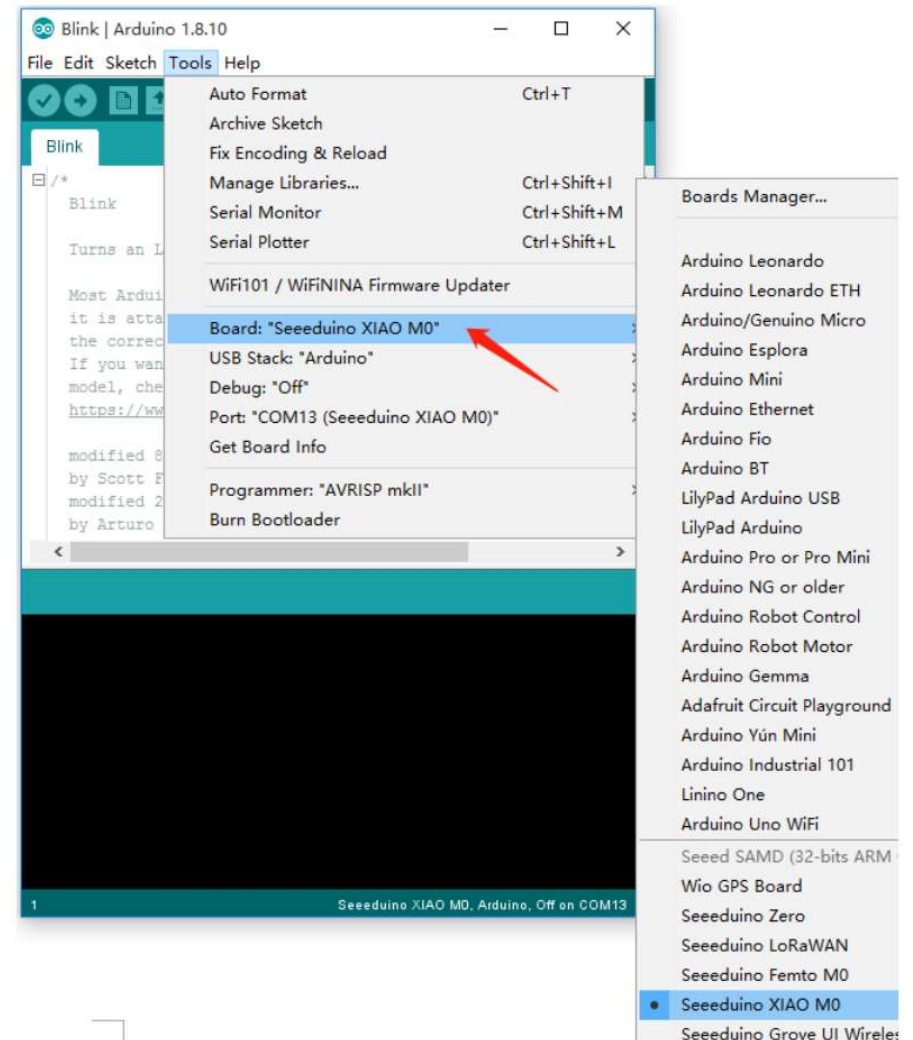


- **výkonný CPU-ARM Cortex-M0 mikrokontroler**
  - 32bit 48MHz SAMD21G18) s 256KB Flash,32KB SRAM, s nízkou spotřebou
  - I/O: 14 GPIO pinů - 11 analogých, 11 digitálních, 1 DAC pin
  - rozhraní 1x I2C, 1x UART, 1x SPI; napájecí a data přes USB-C
- **malá (nejmenší...) velikost – pro nositelné projekty („wearables“)**



# jak přidat do Arduino IDE?

- 1 click on **File > Preference**, fill Additional Boards Manager URLs using this link: [https://files.seeedstudio.com/arduino/package\\_seeduino\\_boards\\_index.json](https://files.seeedstudio.com/arduino/package_seeduino_boards_index.json)
  - 2 click **Tools-> Board-> Boards Manager...**, print keyword "Seeduino XIAO" in the searching blank. Here comes the "Seeed SAMD Boards". Install it.
  - 3 select your board and port - click **Tools-> Board**, find "Seeduino XIAO M0" and select it:
- nyní je možné připojit Xiao pomocí kabelu USB-C
  - a začít tvořit první program
  - tento **postup je obecně platný** i pro jiné Arduino-kompatibilní desky a moduly, které nejsou součástí standardní instalace IDE
  - často se v menu **File > Examples** objeví další příklady programů specifické pro danou desku / modul



# IDE přehled

jednotlivé skupiny Menu zahrnují:

## ■ File

- **New** nové okno IDE se otevře pro vytvoření nového programu
- **Open, Open Recent, Sketchbook** otevření a procházení vlastních dříve uložených programů či projektů, standardně v adresáři pod Documents, lze přesunout
- **Examples** – nabídka příkladů programů využitelných obecně pro jakékoliv moduly, pro konkrétní aktuálně zvolený modul, a pro doinstalované knihovny; cenný zdroj inspirace, případně i kousků kódu ...
- **Close, Save, Save as** – uzavření či uložení programu, vždy ve vlastním adresáři, Filename.**ino**  
starší verze používaly zakončení **xxx.pde**
- **Preferences** – běžná nastavení GUI, umístění souborů, lze zadat, odkud se budou instalovat další komponenty programu pro nové elektronické moduly kompatibilní s Arduino IDE (vytvořené v rámci komunity):

The screenshot displays the Arduino IDE interface. On the right, the 'Examples' menu is open, listing various board-specific examples and libraries. The code editor in the center shows a C++ sketch with a `void loop()` function. The code includes a comment `//turns led on`, an `if (Serial.available())` condition, and a `char input = Serial.read();` statement. The code then checks `if(input == '1')` and `else if(input == '2')` to control an LED. The `else if` branch is currently selected. Below the code editor, there is a message that says 'Invalid library'.

Examples	Keyboard Shortcut
Close	Ctrl+W
Save	Ctrl+S
Save As...	Ctrl+Shift+S
Page Setup	Ctrl+Shift+P
Print	Ctrl+P
Preferences	Ctrl+Comma
Quit	Ctrl+Q

- 03. Analog
- 04. Communication
- 05. Control
- 06. Sensors
- 07. Display
- 08. Strings
- 09. USB
- 10. StarterKit\_BasicKit
- 11. ArduinoISP

Examples for any board

- Adafruit Circuit Playground
- Firmata
- SD
- Stepper
- Temboo
- RETIRED

Examples for chipKIT uC32

- Bridge
- DEIPcK
- DEWFcK
- DSDVOL
- DSPI
- DTWI
- EEPROM
- Ethernet
- Examples
- HTTPServer
- LiquidCrystal
- RAMVOL
- SoftPWMServo
- SoftSPI
- SoftwareSerial
- SPI
- SPIRAMVOL
- Wire

Examples from Custom Libraries

- AD5933
- AD7193
- Adafruit LED Backpack Library
- Adafruit SSD1306
- DHT sensor library
- LMP91000-master
- MCP3221-Library-master
- MD\_AD9833

Additional Boards Manager URLs:



- **Edit** – funkce běžné pro textové editory
- **Sketch** – vytváření a překlad (kompilace) kódu
  - **Verify** - kontrola a **Compile** - kompilace programu
  - **Upload** - jeho zavedení do el. modulu jednoduše přes USB port, nebo speciálním programátorem (např. ICSP rozhraní), **Export** ... binárního (.hex, .bin) souboru pro zavedení do modulu jinak
  - **Include Library** – přidá do projektu potřebnou knihovnu, v rámci kódu v horní části se objeví řádek  
`#include <nazev_knihovny.h>`
  - xxx.h soubory jsou hlavičky definující to, co se dá z dané knihovny xxx.c použít
  - **Add File** – přidání dalšího souboru, např. definice fontů, apod.
- **Tools** – funkce pro zavedení programu, sledování jeho běhu
  - **Manage Libraries** – práce s interními i přidanými knihovnami
  - **Serial Monitor** a **Serial Plotter** – textový výstup z modulu připojeného přes USB port, pokud má povahu řady čísel, tak lze zobrazit i jako časový graf
  - **Board: „deska“** - je zobrazena aktuálně vybraná Arduino deska či modul (board) a někdy i další pro ni relevantní funkce, je možné zvolit standardní a další přidané moduly – ty lze přidávat pomocí **Boards Manager** submenu – pokud byl přidán relevantní link v rámci Preferences, tak tady se objeví zahrnuté moduly
  - **Get Board Info** – přečtou se detaily zevnitř modulu
  - **Ports** – kudy se bude zavádět zkompilevaný program, ve Windows je zde nabídka nalezených sériových portů – vybere se ten správný; (viz i Device Manager – Ports – pokud není jisté, který byl zvolen systémem při připojení modulu k počítači)



# IDE toolbar

- rychlý přístup pro nejčastěji používané příkazy z Menu



- **Text Editor** - v rámci textového okna se pak píše vlastní kód pro daný program nebo jinou složku aktivního projektu (nová knihovna, apod.)
  - každý projekt se vždy otevírá v rámci samostatného nového okna IDE
  - probíhá zvýraznění částí textu dle syntaxe pro lepší přehlednost
- **Output Pane** – dolní tmavá textová oblast, vypisují se zde různé provozní údaje a informace
  - průběh kompilačního procesu
  - chyby znemožňující kompilaci programu – třeba odstranit
  - scházející knihovny
  - komunikace s deskou

# Základy programování

- pokud se v rámci IDE zadá vytvoření nového programu, tak se v textovém editoru objeví:
- tyto dvě funkce reprezentují nutné minimum každého programu
- v rámci **setup()** se provádí jednorázové inicializační úkony, definování hodnot, otevření komunikačních linek, nastavení funkce jednotlivých pinů, apod.
  - provede se pouze jednou po stisku Reset, nebo po připojení napájení
- naopak funkce **loop()** a v ní obsažené příkazy se vykonává pořád dokola po celý běh zavedeného programu (až do Reset, nebo odpojení napájení)
  - provádí se zde např. čtení dat ze vstupů a nastavování výstupů – digitálních či analogových, ovládání periferních modulů – opět nastavování či čtení dat,
  - realizuje se zpracování a vyhodnocování dat
  - výsledky se posílají do vnějšího prostředí (počítač, internet, ...) a reaguje se na odtud přicházející data
  - zatím pomineme další možnosti dané reakcí modulu na **interrupt** - přerušení vyvolaná různými hw událostmi

```
void setup() {  
    // put your setup code here, to run once:  
}  
  
void loop() {  
    // put your main code here, to run repeatedly:  
}
```

# první program - Blink

- bývá v [Examples ... 01. Basics](#), účelem je ověřit funkčnost nově připojeného modulu

```
/*
  Blink
  Turns an LED on for one second, then off for one second, repeatedly.
  Most Arduinos have an on-board LED you can control. On the UNO, MEGA and ZERO
  it is attached to digital pin 13, on MKR1000 on pin 6. LED_BUILTIN is set to
  the correct LED pin independent of which board is used.
*/

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(1000); // wait for a second
  digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the voltage LOW
  delay(5000); // wait for a second
}
```

- v případě XIAO modulu žlutá uživatelská LEDka svítí naopak při stavu LOW na pinu
- výpisy programů jsou standardně z důvodu zarovnání (indentation) textu prezentovány fontem s fixní šířkou znaků (Courier New) – významně to vylepšuje čitelnost kódu



# program Hello World

- demonstrece komunikace s okolím přes USB seriový port

```
/*
  AnalogReadSerial
  Reads an analog input on pin A0, prints the result to the Serial Monitor.
  Graphical representation is available using Serial Plotter (Tools > Serial Plotter
  menu) .
  Attach the center pin of a potentiometer to pin A0, and the outside pins to +3.3V and
  ground.
*/

void setup() {
  Serial.begin(9600); // initialize serial communication at 9600 bits per second:
  Serial.println("Hello World!"); // let know that you have started successfully
}

void loop() {
  int sensorValue = analogRead(A0); // read the input on analog pin 0:
  Serial.println(sensorValue); // print out the value you read:
  delay(500); // delay (ms) in between reads for stability
}
```

- textový výstup lze sledovat v Serial Monitoru, v Serial Plotteru se bude generovat graf dle přijatých číselných hodnot
- potenciometr není nutný, hodnoty lze měnit dotykem pinu A0 prstem...

- je to místo, kde se uchovávají data; každá má vlastní jméno, datový typ a hodnotu
  - // značí že další část je komentář a nemá vliv na program
  - jednotlivé příkazy se ukončí znakem ; (středník)

```
// deklarace promenne x, typ je integer
int x;
// prirazeni hodnoty
x = 10;
// tyto dve operace se daji spojit do jedne
int y = 10;
```

- pokud (**globální**) proměnnou definujeme na počátku programu před definováním funkcí (vč. setup a loop), lze ji využít kdekoliv
- definice (**lokální**) proměnné uvnitř funkce omezuje použití pouze v rámci dané funkce – uvnitř **bloku** vymezeného složenými závorkami { příkazy ... }
- **názvy** – mohou obsahovat malá a velká písmena, číslice (ne na prvním místě) a podtržítka, ne klíčová slova

# datové typy

## ■ číselné

- **byte** – uchovává přirozená čísla 0 až 255, velikost 1 byte resp. 8 bitů, jinak též reprezentuje jeden ASCII znak
- **integer** – celá čísla od -32768 do 32767, velikost 2 byte / 16 bit, může se lišit dle typu procesoru (8 bit AVR vs. 32 bit ARM)
- **long** – 4 byte / 32 bit celá čísla, -2 147 483 648 do 2 147 483 647
- **float** – desetinná čísla od  $-3.4028235 \cdot 10^{38}$  do  $3.4028235 \cdot 10^{38}$   
pozn. bude se používat desetinná tečka – kompatibilita s angl. textem a zdrojovými kódy

## ■ logické

- **boolean** – pouze dvě hodnoty true nebo false  
(alternativní význam je 1 nebo 0, HIGH nebo LOW)

## ■ znakové

- **char** – uchovává jeden znak (character) jako jeho číselnou ASCII reprezentaci  
`char c = 'A'; char d = 65; // ASCII hodnota písmena A`  
znaky se zadávají pomocí jednoduchých uvozovek

## ■ pole array je souborem proměnných stejného typu

- přístupných pomocí **[indexu]** (od 0), příklady:

```
int myInts[6]; int myPins[ ] = {2, 4, 8, 3, 6};
```

```
int mySensVals[6] = {2, 4, -8, 3, 2}; char message[6] = "hello";
```

## Data Types

array

bool

boolean

byte

char

double

float

int

long

short

size\_t

string

String()

unsigned char

unsigned int

unsigned long

void

word



# texty (string, řetězec znaků)

- dvě možnosti reprezentace – jako **pole znaků** (null-terminated – zakončené znakem s ASCII kódem 0, resp. ‘\0’ jako znak), nebo pomocí datového **typu String** (více možností, manipulační funkce)
- **char pole[ ] ukázky:**
  - `char str1[15]; char Str2[8] = {'a', 'r', 'd', 'u', 'i', 'n', 'o'};`
  - `char Str3[8] = {'a', 'r', 'd', 'u', 'i', 'n', 'o', '\0'};`
  - `char Str4[ ] = "arduino"; char Str5[8] = "arduino"; char Str6[15] = "arduino";`
  - text se vždy ohraničuje pomocí dvojitého uvozovky
- **možné String deklarace:**
  - `String s1 = "Hello String"; // using a constant String`
  - `String s2 = String('a'); // converting a constant char into a String`
  - `String s3 = String(s2 + " with more"); // concatenating two strings`
  - `String s4 = String(13); // using a constant integer`
  - `String s5 = String(analogRead(0), DEC); // using an int and a base`
  - `String s6 = String(45, HEX); // using an int and a base (hexadecimal)`
  - `String stringOne = String(5.698, 3); // using a float and the decimal places`
- viz i **Examples – 08.Strings** pro příklady přidružených funkcí

# konstanty

- předdefinované proměnné s neměnnou hodnotou, výhodné při opakovaném používání, zpřehlednění kódu
- **false** / **true** ... logické, odpovídají 0 a 1 (resp. nějaký integer různý od nuly ... 1, 55, -200)
- **HIGH** / **LOW** ... napěťové úrovně dig. pinů dané i pracovním napětím mikrokontroleru - pro 5V je HIGH od 3.0V nahoru, LOW od 2.0 V dolů (může být lehce modifikováno – viz manuály k čipům)
- **LED\_BUILTIN** ... reprezentuje číslo dig. pinu, na který je připojena uživatelem použitelná LED (obvykle pin 13)
- **integer** konstanty ... lze zadat různě s využitím čísel ze soustav: decimálních (-5, 11, 1254 - běžné), **hexadecimálních** (0x7F, 0xFF31, z číslic a písmen A..F, a..f, **0x** je nutný prefix) a **binárních** (0b1111011, z číslic 0 a 1, **0b** je nutný prefix)
  - `n = 0x101; // shodné jako n = 257; odpovídá ((1 * 16^2) + (0 * 16^1) + 1)`
  - někdy se za číslo přidá `u` nebo `U` pro vnucení „unsigned“ typu, také lze `l` nebo `L` pro vnucení „long“ typu
- desetinná čísla lze zadat s desetinnou tečkou `x = 10.32`
  - je možné použít vědecké vyjádření s `e` nebo `E` jako exponent, `y = 95e-12`

# digitální I/O

- **pinMode**(pin, mode) nakonfiguruje - typicky v rámci setup() bloku daný pin jako vstup nebo výstup
  - pin je číslo konkrétního pinu – 0, 1, ..., i analogové vstupy – A0, A1, ...
  - mode zahrnuje možnosti INPUT, INPUT\_PULLUP (interní připojení na napájecí napětí přes 20 kOhm odpor) a OUTPUT
- **digitalWrite**(pin, value) nastavuje hodnotu na HIGH nebo LOW
- **digitalRead**(pin) přečte stav daného pinu; pokud není k ničemu připojen, vrací se náhodná hodnota

```
int ledPin = 13; // LED connected to digital pin 13
int inPin = 7; // pushbutton connected to digital pin 7
int val = 0; // variable to store the read value

void setup() {
  pinMode(ledPin, OUTPUT); // sets the digital pin 13 as output
  pinMode(inPin, INPUT); // sets the digital pin 7 as input
}

void loop() {
  val = digitalRead(inPin); // read the input pin
  digitalWrite(ledPin, val); // sets the LED to the button's value
}
```



# analog I/O

- **analogRead(pin)** přečte velikost napětí na specifikovaném vstupu
  - pin může být A0 až A5 téměř u všech desek, někdy i další hodnoty
  - rozlišení je běžně 10 bit, tj. vrací hodnotu mezi 0 a 1023 pro napětí mezi 0 až 5 V
  - u „lepších“ modulů 12 bit (0 až 4095) – je možné nastavit tuto lepší (pomalejší...) přesnost čtení pomocí **analogReadResolution(bits)**, bits jsou 10 nebo 12
  - rozlišení lze jednoduše vylepšit opakovaným čtením daného vstupu (pokud měřená hodnota není zcela konstantní)
- **analogReference(type)** nastavuje referenční napětí
  - vůči kterému se porovnává hodnota čtená **analogRead** funkcí; toto napětí pak reprezentuje maximální vracenou bitovou hodnotu (1023)
  - **DEFAULT**: pracovní napětí modulu 5 V nebo 3.3 V dle typu modulu
  - **INTERNAL**: vnitřní referenční napětí, např. 1.1 V u ATmega328P, 2.56 V u the ATmega32U8
  - **EXTERNAL**: vnější stabilní napětí přivedené na AREF pin (mezi 0 a 5V)
  - odlišné konstanty a možnosti jsou pro moderní SAMD moduly
- **analogWrite(pin, value)** dvojí význam, běžně na pinu generuje PWM
  - „pulse width modulation“ – obdélníkové pulzy s proměnlivou dobou trvání a amplitudou 5 V – takto se vlastně mění výstupní „energie“ – třeba pro LED
  - moduly MKR řady, XIAO a Nano 33 BLE mají ale i „pravý“ D/A převodník na jednom pinu (DAC0) – a na tom se pak objeví napětí dle zadané int hodnoty

# příklady čtení a nastavení

```
// sledování vstupu, po sériové lince (na Serial Monitor / Plotter):  
int analogPin = A3; // výstup z potenciometru mezi 0 a 5 V  
int val = 0;  
  
void setup() { Serial.begin(9600); } // setup serial  
  
void loop() {  
    val = analogRead(analogPin); // read the input pin  
    Serial.println(val); // debug value  
}
```

```
// nastavení výstupu pro LED dle hodnoty potenciometru  
int ledPin = 13; // LED connected to digital pin 13  
int analogPin = 3; // potentiometer connected to analog pin 3  
int val = 0; // variable to store the read value  
  
void setup() { pinMode(ledPin, OUTPUT); } // sets the pin as output  
  
void loop() {  
    val = analogRead(analogPin); // read the input pin  
    analogWrite(ledPin, val / 4);  
    // analogRead values go from 0 to 1023, analogWrite from 0 to 255  
}
```

# čas a pauzy

- **delay(ms)** zastaví běh programu na daný počet milisekund
  - zastaví se opravdu skoro všechno – nějaká událost se tedy může „ztratit“
  - výjimkou jsou pouze přerušení
  - není to třeba řešit u velmi jednoduchých programů
  - příklad lepšího čekání je v Examples – 02.Digital – [BlinkWithoutDelay](#)
- **delayMicroseconds(us)** obdoba v mikrosekundách
  - rozumné dávat hodnoty do 16 383, pak to může „přetéct“
- **millis()** vrací počet milisekund od započetí běhu programu
  - unsigned long, přeteče po 50 dnech; není to tedy dobré číst do int time = millis(), protože int typ nemá dostatečnou kapacitu
  - lze takto sledovat uplynutí daného intervalu bez zablokování programu
- **micros()** obdoba v mikrosekundách, přeteče po 70 minutách
  - rozlišení je 4 us u desek běžících na 16 MHz, u lepších (=rychlejších) pak 1 us

# test podmínky

- **if (else)** pokud je splněna určitá podmínka, vykoná se následující příkaz, jinak se přeskočí a nebo se vykoná příkaz po else:

```
if (condition) {  
  //statement(s)1  
}
```

```
if (x>120) digitalWrite(pin, HIGH);  
  
if (x>150) {  
  digitalWrite(LEDpin1, HIGH);  
  digitalWrite(LEDpin2, HIGH);  
}
```

```
if (condition1) {  
  // do Thing A  
}  
else if (condition2) {  
  // do Thing B  
}  
else {  
  // do Thing C  
}
```

else if není nezbytné, nebo se klidně může použít vícekrát příkaz po poslední else se vykoná, když nic před tím nebylo splněno

- možné **podmínky**:
- $x == y$  (x is equal to y) // pozor, jedno = nestačí!
- $x != y$  (x is not equal to y)
- $x < y$  (x is less than y)                       $x > y$  (x is greater than y)
- $x <= y$  (x is less than or equal to y)     $x >= y$  (x is greater than or equal to y)
- lze použít i test na číselnou hodnotu, nenulová se bere jako TRUE



# Co se může pokazit ...

- **Arduino Board is not Recognized**
  - problém s připojením přes USB – schází drivery – nainstalovat a někdy i restartovat PC, špatně zvolený komunikační port v IDE, levný čínský klon
- **Unable to Upload Code**
  - v IDE je špatně zvolený typ desky, někdy třeba i zvolit bootloader či rychlost zavádění programu (zpomalit...), restartovat IDE
- **Board Not in Sync**
  - program zaveden, ale něco se pokazilo – není něco na pinech 0,1?, resetovat desku, odpojit a znovupřipojit desku, restartovat IDE. Prohodit desku s jinou shodnou, prohodit PC
- **Code Fails to Start on Power Reset**
- **Unable to Set Host COM Port Automatically**

# Co se může pokazit ...

- **Invalid Device Signature**
- **Launch4j Error**
- **Serial Port Already in Use**
- **Uploads Successful but No Effect**
- **Sketch Too Large**
- **Java.lang.StackOverflow**
- **Missing Libraries or Header Files**

## **Další informace**

- **Jason Hamilton: Arduino programming a complete beginners' guide on learning to engineer and program Arduino. 2020, 93 stran**
  - povrchně zpracované příklady, ale srozumitelná prezentace obecných principů
- **anon (TinkerGen):  
seeeduino\_XIAO\_in\_Action\_Minitype\_Wearable\_Projects\_Step\_by\_Step 2021, 132 stran**
  - zaměřeno konkrétně na Xiao modul
- **Steven F. Barrett: Arduino III Internet of Things. 2021, 237 str.**
  - jak s arduinem do internetu věcí