Jaculus

JavaScript/Typescript pro ESP32

Petr Kubica, FI MUNI
github.com/cubicap

Kdo?

- Petr Kubica (cubicap)
  - FI MUNI
- Robotika Brno
  - Komunita nadšenců do robotiky, programování…
  - Robotárna (SVČ Helceletka)

Programování mikrokontrolérů

- Kompilované jazyky – C, C++, Rust (...?)
- Složité a pracné
- Velká zodpovědnost programátora
- Komplexní vývojové nástroje
- Dlouhé iterace změny kódu

Jak potom … učit začátečníky?

… protypovat?

Použijme vysokoúrovňový jazyk – třeba JavaScript

- Oddělení ovladačů od aplikační logiky
- Vysoká úroveň abstrakce
- Jednodušší vývojové nástroje
- Rychlá iterace kódu
- Asynchronní, událostmi řízené programování

Jaculus

- Firmware pro mikrokontrolér
  - ESP32-S3, (ESP32, ESP32-C3)
- Aplikace pro počítač programátora
  - Prvotní nahrání firmwaru/aktualizace
  - Nahrávání kódu
  - Zobrazení výstupu programu
  - Překlad TypeScriptu

Co Jaculus umí

- Spustit JavaScriptový program na ESP32
- Ovládat základní periferie
  - GPIO, ADC, I2C, PWM, ...
  - WS2812 LED, DC motory
  - UDP soket
  - ...
- Připojení přes sériový port nebo WiFi (TCP soket)

Napišme kousek kódu...
Třeba rozblikejme LEDku!

```
1   import * as gpio from "gpio";
```

```
1   import * as gpio from "gpio";
2
3   const LED_PIN = 45;
4   gpio.pinMode(LED_PIN, gpio.PinMode.OUTPUT);
```

```
import * as gpio from "gpio";

const LED_PIN = 45;
gpio.pinMode(LED_PIN, gpio.PinMode.OUTPUT);

let state = false;
setInterval(() => {
    gpio.write(LED_PIN, state ? 1 : 0);
    state = !state;
}, 1000);
```

```
import * as gpio from "gpio";

const LED_PIN = 45;
gpio.pinMode(LED_PIN, gpio.PinMode.OUTPUT);

let state = false;
setInterval(() => {
    gpio.write(LED_PIN, state ? 1 : 0);
    state = !state;
}, 1000);

const LED_PIN2 = 46;
gpio.pinMode(LED_PIN2, gpio.PinMode.OUTPUT);

let state2 = false;
setInterval(() => {
    gpio.write(LED_PIN2, state2 ? 1 : 0);
    state2 = !state2;
}, 750);
```

```
import * as gpio from "gpio";

const LED_PIN = 45;
gpio.pinMode(LED_PIN, gpio.PinMode.OUTPUT);

let state = false;
setInterval(() => {
    gpio.write(LED_PIN, state ? 1 : 0);
    state = !state;
}, 1000);

const LED_PIN2 = 46;
gpio.pinMode(LED_PIN2, gpio.PinMode.OUTPUT);

let state2 = false;
setInterval(() => {
    gpio.write(LED_PIN2, state2 ? 1 : 0);
    state2 = !state2;
}, 750);

const BUTTON_PIN = 18;
gpio.pinMode(BUTTON_PIN, gpio.PinMode.INPUT_PULLUP);
gpio.on("falling", BUTTON_PIN, () => {
    console.log("Button pressed!");
});
```

Jak by totéž mohlo vypadat v C++?

```
1   #include <Arduino.h>
```

```
#include <Arduino.h>

constexpr uint8_t LED_PIN = 45;

void setup() {
    pinMode(LED_PIN, OUTPUT);
}
```

```cpp
#include <Arduino.h>

constexpr uint8_t LED_PIN = 45;

void setup() {
    pinMode(LED_PIN, OUTPUT);
}

void loop() {
    digitalWrite(LED_PIN, HIGH);
    delay(1000);
    digitalWrite(LED_PIN, LOW);
    delay(1000);
}
```

Jak si můžete vyzkoušet Jaculus i vy?

www.jaculus.org/getting-started

```
petr@pop:~$ node --version
v22.14.0
petr@pop:~$ npm install -g jaculus-tools@latest

changed 73 packages in 1s

17 packages are looking for funding
  run `npm fund` for details
petr@pop:~$
```

```
petr@pop:~$ npx jac
Usage: jac <command>

Tools for controlling devices running Jaculus

Commands:
  help            Print help for given command
  version         Get version of device firmware
  list-ports      List available serial ports
  install         Install Jaculus to device
  build           Build TypeScript project
  flash           Flash code to device (replace contents of ./code)
  pull            Download a file/directory from device
  ls              List files in a directory
  read            Read a file from device
  write           Write a file to device
  rm              Delete a file on device
  mkdir           Create a directory on device
  rmdir           Delete a directory on device
  upload          Upload a file/directory to device
  format          Format device storage
  project-create  Create project from package
  project-update  Update existing project from package skeleton
  resources-ls    List available resources
  resources-read  Read a resource from device
```

Jaculus web installer

A Serial Flasher utility for Jaculus - ESP32

View Jaculus Documentation | View Source Code

Program

Baudrate:

Connected to device: ESP32-S3

921600

Select chip: ESP32-S3

Select variant: Generic Quad PSRAM

Select Jaculus version: 0.0.21

Erase Flash: No

Flash | Copy Trace

Disconnect

```
esptool.js
Serial port
Connecting....
Detecting chip type... ESP32-S3
Chip is ESP32-S3
Features: Wi-Fi,BLE
Crystal is 40MHz
MAC: 34:85:18:51:51:38
Uploading stub...
Running stub...
Stub running...
Warning: Image file at 0x0 doesn't look like an image file, so not changing any flash settings.
Compressed 21072 bytes to 13055...
Writing at 0x0... (100%)
Wrote 21072 bytes (13055 compressed) at 0x0 in 0.541 seconds.
Compressed 3072 bytes to 104...
Writing at 0x8000... (100%)
Wrote 3072 bytes (104 compressed) at 0x8000 in 0.067 seconds.
```

File Edit Selection View Go Run Terminal Help

ts-examples

EXPLORER

TS-EXAMPLES

- @types
- build
- src
  - TS adc.ts
  - TS blink.ts
  - TS blinksmart.ts
  - TS echo.ts
  - TS filesystem.ts
  - TS gomoku.ts
  - TS index.ts
  - TS ledc.ts
  - TS piezo.ts
  - TS radioblink.ts
  - TS radioecho.ts
  - TS readline.ts
  - TS repl.ts
  - TS servo.ts
  - TS snake.ts
- README.md
- tsconfig.json

OPEN EDITORS

OUTLINE

TIMELINE

blinksmart.ts | radioecho.ts | echo.ts | snake.ts | radioblink.ts | adc.ts | gomoku.ts | simpleradio.d.ts | index.ts

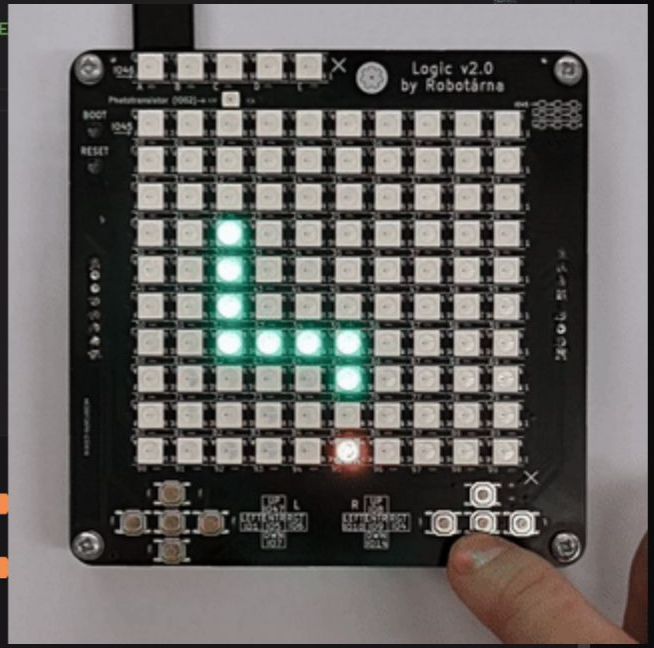
src > TS snake.ts > ...

```ts
import { SmartLed, Rgb, LED_WS2812 } from "smartled";
import * as gpio from "gpio";

/**
 * A simple Snake game.
 * Implemented for the Logic 1.1 with ESP32, and the Logic 2.0 prototype with E
 * (https://github.com/RoboticsBrno/RB3205-Logic)
 */

if (PlatformInfo.name == "ESP32") {
    var LED_PIN = 23;
    var POWER_PIN = 16;

    var UP = 14;
    var DOWN = 26;
    var LEFT = 32;
    var RIGHT = 13;
    var MIDDLE = 17;
}
else if (PlatformInfo.name == "ESP32-S3") {
    var LED_PIN = 45;
    var POWER_PIN = 0;
```



TERMINAL OUTPUT DEBUG CONSOLE PROBLEMS

```
Petr@LENOVO-PETR ~\Documents\Skolni\SBAPR\Jaculus\ts-examples master ≡ ?2 ~3 2
$

Petr@LENOVO-PETR ~\Documents\Skolni\SBAPR\Jaculus\ts-examples master ≡ ?2 ~3 2
$ npx jac build flash monitor --port COM9
Compiled successfully
Connecting to serial at COM9 at 921600 bauds... Connected.

info:    Getting current data hashes
info:    Uploading build/index.js to code/index.js
info:    Files synced, 1 uploaded, 0 deleted
info:    Device: Starting machine
Score: 1
Score: 2
Score: 3
```

COM9 Build Flash Monitor Build, Flash and Monitor ⊗ 0 ⚠ 0 0 Live Share Git Graph

Ln 9, Col 1 Spaces: 4 UTF-8 LF {} TypeScript 1.3 hrs 100%

Závěr

- Aktivně používaný na výuku a prototypování
- Stále ve vývoji
  - Grafické programování (Blockly/Scratch)
  - Quality of life
  - Více knihoven k periferiím (poptávkou řízené)