Visual Studio Code

Petr Novák / novakpe@cvut.cz / 2022-01-18

Obsah

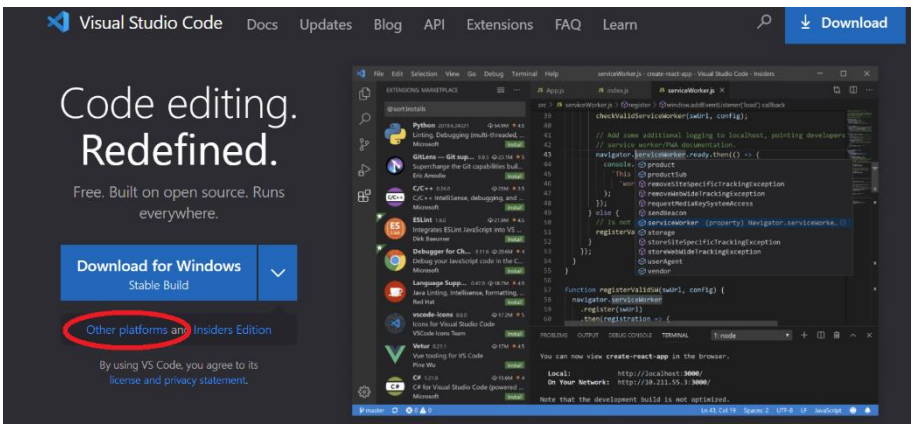
1 Úvod

Tento popis nemusí být úplně správný. Pro činnost Visual-Studio-Code na Linuxu je potřeba instalace „.NET Core“ což je popsáno v jiném dokumentu.

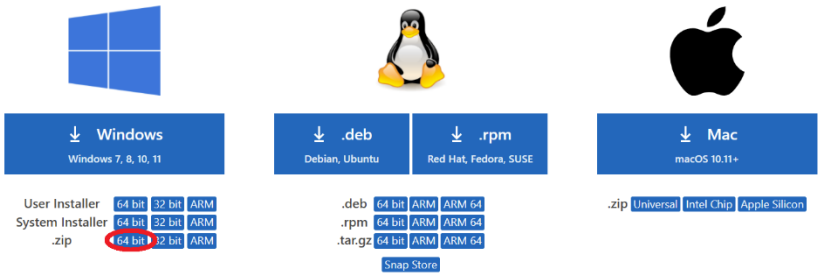
2 Instalace Visual Studio Code

Visual Studio Code není (asi) vždy potřeba instalovat, stačí stáhnout tzv „portable verzi“, tu pouze nakopírovat a spustit (samozřejmě jej lze instalovat například pro snadnější spuštění / aktualizaci).

- Stáhnout požadovaný typ (podle instalace) ze stránky: „https://code.visualstudio.com/“
- Na úvodní stránce vybrat „Other Platforms“



- Pro Microsoft/Windows postačuje typ ZIP (podle verze operačního systému 32/64bits). Udělat adresář (například podle stažené verze VSC) a tam ZIP rozbalit (ZIP bohužel neobsahuje adresář).



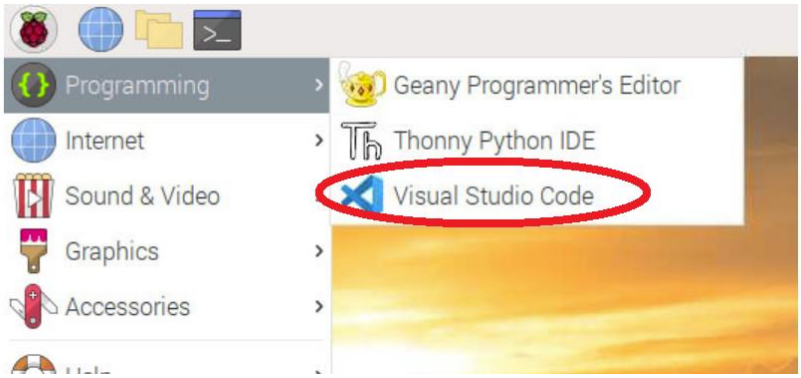
- Visual Studio Code se spouští pomocí „VSCode-win32-x64-1.63.2/Code.exe“ (jak na Linuxu – nevím, tam jsem jej instaloval).
- Na Linuxu (Raspberry PI 4) jsem Visual Studio Code instaloval pomocí (z terminálu):

  sudo apt update
  sudo apt install code

  Pozdější aktualizace by se měla vykonávat pomocí (z terminálu):

  sudo apt update
  sudo upgrade code

  Instalované Visual Studio Code se spouští z nabídky aplikací:



3 Vytvoření projektu

- Nastavit se do adresáře, pod kterým bude vytvořen pod-adresář s novým projektem (nový projekt bude vytvořen / umístěn teprve až ve vytvořen pod-adresáři).
- Spustit příkazový řádek / terminál (na Linuxu Ctrl+Alt+T) a zadat:

  dotnet new console --framework net6.0 -o NazevProjektu
  dotnet new webapi …
  dotnet new classlib …
  dotnet new xunit …

- Vytvoří se pod-adresář „NazevProjekt“ a v něm základ nového projekty

Poznámky:

- Vytvořený projekt v prostředí MS/Windows lze celý nakopírovat na Linux (Raspberry PI 4) a tam bez jakýchkoli úprav použít.

4 Otevření projektu

- Spustit Visual Studio Code
- File (menu v záhlaví) – Open Folder … (položka v rozbalovacím menu)
- V seznamu adresářů označit (pod)adresář v němž je požadovaný projekt umístěn. Případně vlézt do tohoto (pod)adresáře (zřejmě je použitelné oboje).
- Zobrazí se seznam souborů / adresářů patřících otevřenému projektu
- Pokud nebude seznam souborů / adresářů zobrazen, tak kliknout na ikonu představující dva překrývající se listy u levého horního rohu Visual Studio Code.

Zavření projektu je pomocí **File** (menu v záhlaví) a **Close Folder** (položka v rozbalovacím menu).

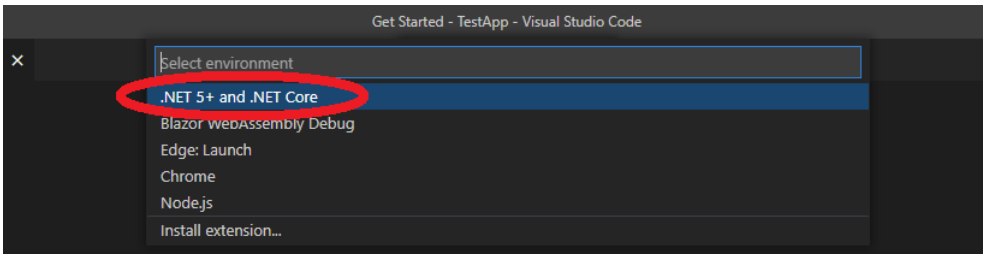
Po instalaci Visual Studio Code a otevření prvního projektu, případně kliknutí / zobrazení souboru „Program.cs“ se někde ve VisualStudioCode zobrazí informace o možnosti instalace rozšíření pro C# (OmniSharp), tak tuto instalaci povolit / vykonat.

5 Kompilace a spuštění z konzole

- Otevřít terminál: Terminál (menu v záhlaví) – New Terminál (položka v rozbalovacím menu)
- Kompilace projektu: dotnet build
- Spuštění projektu: dotnet run

6 Spuštění a ladění

Při prvním spuštění ladění pomocí **Run** (menu v záhlaví) a **Start Debugging** (položka v rozbalovacím menu) se zobrazí v horní části Visual Studio Code dotaz na nastavení tohoto ladění formou:



Zvolit položku „.NET 5+ and .Net Core“. Vytvoří / zobrazí se soubor „launch.json“. Ten je vhodný pro nastavení ladění aplikace (uvedeno dále).
Nyní lze do programu vložit „break point“ a pomocí **Run** (menu v záhlaví) a **Start Debugging** (položka v rozbalovacím menu) spouštět program v ladicím režimu. Iony pro krokování jsou ve středu horní části Visual Studio Code.

Poznámky:

- Pokud je potřeba konzolové aplikace předat při ladění nějaké parametry lze je vložit do „launch.json“ do řádku s "args": [] například následovně
  "args": [„PrvniParametr“, „DruhyParametr“, „TretiParametr“, „CtvrtyParametr“],

7 Distribuce hotové aplikace

Po vytvoření a odladění aplikace je potřeba vykonat „Publish“, v podstatě tvar aplikace pro její nasazení na cílové místo a to následovně:

- Ve Visual Studio Code spustit terminal.
- Zadat: dotnet publish --configuration Release
- V adresáři projektu „bin/Release/net6.0/publish“ jsou soubory pro distribuci aplikace.

Soubor „NazevProjektu.pdb“ lze odstranit, obsahuje nějaké minimální informace pro ladění i v případě nasazení aplikace.

8 Spuštění hotové aplikace

Na MS/Windows lze hotovou aplikaci spouštět následovně:

NazevProjektu.exe PrvniParametr DruhyParametr …

Na Linuxu nebo i zcela obecně lze aplikaci spouštět následovně

dotnet NazevProjektu.dll PrvniParametr DruhyParametr …

Poznámky:

- Mezi soubory aplikace je soubor s názvem „NazevAplikace.exe“ sloužící pravděpodobně pouze jako soubor pro zjednodušené spouštění aplikace v prostředí MS/Windows. Skutečná aplikace se nachází v souboru „NazevAplikace.dll“.

9 Soubory projektu

Program.cs – Výchozí soubor aplikace obsahující funkci „main“.
NazevProjektu.csproj – Projektový soubor pro VisualStudioCode.

.vscode/Launch.json – Nastavení pro spouštění aplikace (zejména pro ladění).

10 Projekt společný pro více OS

.NET Core je multiplatformní prostředí a v mnoha případech se bude vytvářet projekt, který bude používán na více operačních systémech / zařízeních současně. Bohužel vždy se však najdou části kódu, které se budou trochu lišit (v současné době například přehrávání zvuku).
Ve zdrojových souborech lze použít snadno podmíněný překlad, například:

#if WINDOWS

```
  // vysloveni textu
  XUtils.Speech.TextToSpeech.Speak(text);
#endif
#if RPI
  // prehrani tohoto souboru externi aplikaci a cekani na dokonceni
  player.Play(Path.Combine(Project.App.AppConfigMain.appDirBase, "Sounds", text + ".mp3")).Wait();
#endif
```

Symboly k podmíněnému překladu lze ve Visual Studio Code definovat v projektovém souboru „NazevProjektu.csproj“ například takto:

```
<Project Sdk="Microsoft.NET.Sdk">
  <PropertyGroup>
    <OutputType>Exe</OutputType>
    <TargetFramework>net6.0</TargetFramework>
    <ImplicitUsings>enable</ImplicitUsings>
    <Nullable>disable</Nullable>
    <DefineConstants>RPI</DefineConstants>
  </PropertyGroup>
</Project>
```

V podstatě zcela stejně i ve Visual Studio (Community na Windows) a to rovněž v projektovém souboru „NazevProjektu.csproj“ zadání například:

```
<Project Sdk="Microsoft.NET.Sdk">
  <PropertyGroup>
    <OutputType>WinExe</OutputType>
    <TargetFramework>net6.0-windows</TargetFramework>
    <Nullable>disable</Nullable>
    <UseWPF>true</UseWPF>
    <DefineConstants>WINDOWS</DefineConstants>
  </PropertyGroup>
…
```

11 Použití GPIO pinů (např. Raspberry PI 4)

- Ve 'Visual Studio Code' naistaloval Nuget Gallery
  - Poslední ikona představuící čtyři čtverečky (jeden odsazený) ve sloupci vlevo.
  - Zobrazí se seznam 'Extension(s) …'.
  - Najit 'Nuget Gallery' a ten instalovat.
- Instalovat rozšíření pro GPIO
  - Ctrl + Shift + P zobrazí se nahoře seznam (něčeho) ...
  - Najit 'Nuget: Open Nuget Gallery' a spustit.
  - Hledat 'System.Device.GPIO' a instalovat.

Příklad použití:

```
using System;
```

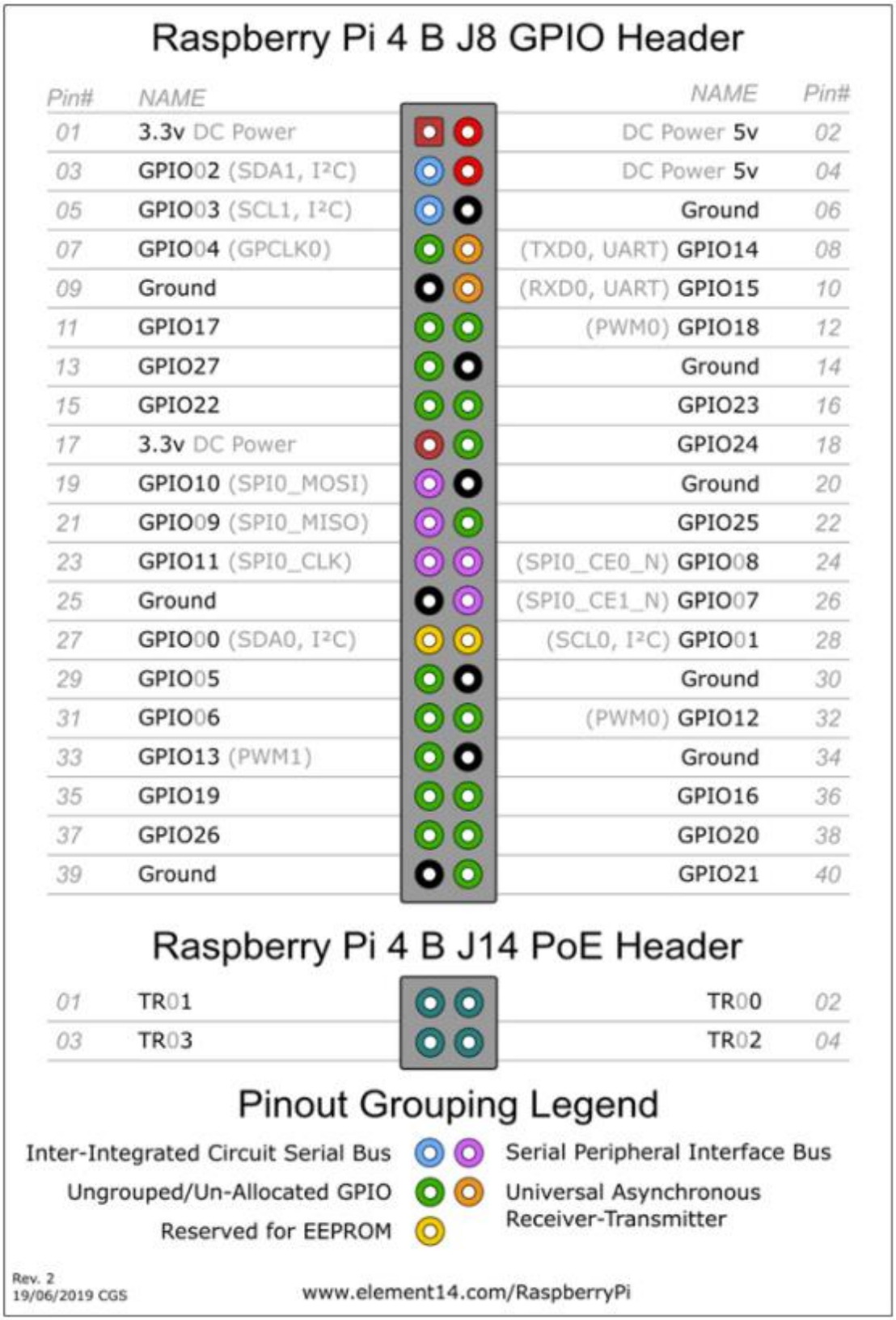
```
using System.Device.Gpio;
using System.Threading;

// (LED je mezi „pin18 = GPIO 24“ a „pin20 = GND“)
int ledPin = 24; //GPIO24 is pin 18 on RPi
int ledOnTime = 1000; // led on time in ms
int ledOffTime = 500; // led off time in ms

using GpioController controller = new();
controller.OpenPin(ledPin, PinMode.Output);

while (true)
{
   controller.Write(ledPin, PinValue.High);
   Thread.Sleep(ledOnTime);

   controller.Write(ledPin, PinValue.Low);
   Thread.Sleep(ledOffTime);
}
```



Návod na použití je například zde (po WWW je jich mnoho):
https://wellsb.com/csharp/iot/raspberry-pi-gpio-csharp-led