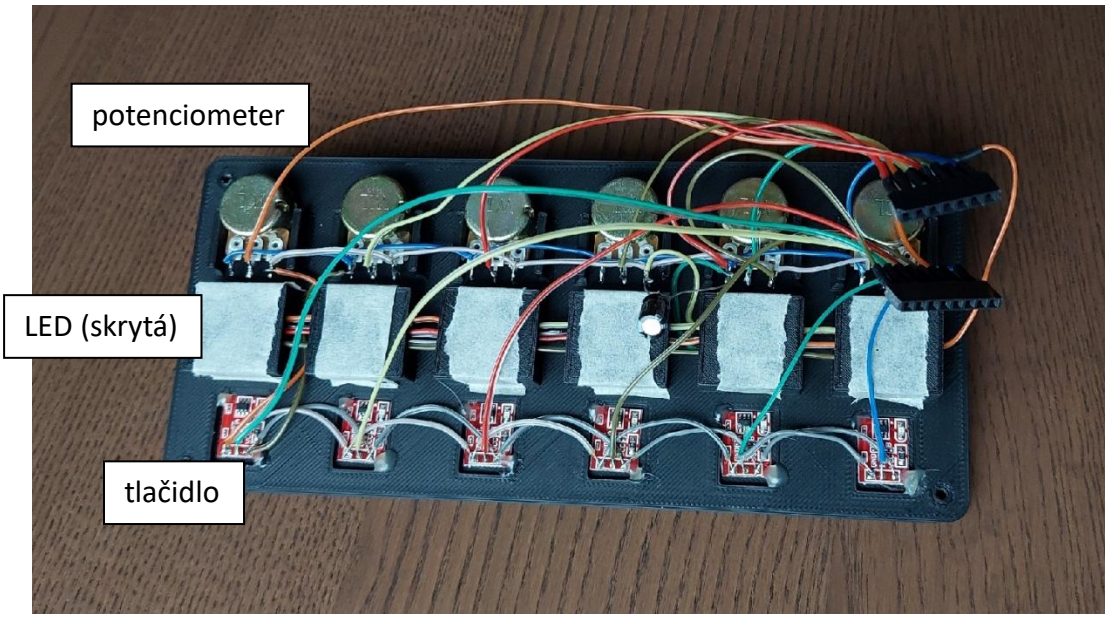
Report zo zimného semestra

Hardvér

Ovládač hlasitosti má 6 samostatných kanálov, ktoré sú konfigurovateľné pomocou softvéru, ktorý bol náplňou projektu na zimný semester. Na pripojenie k počítaču slúži USB C konektor. Jadrom ovládača je doska Arduino Nano, ktorá číta hodnoty z potenciometrov a tlačidiel, ovláda LED diódy a riadi komunikáciu s PC cez sériový port. Každý kanál pozostáva z potenciometra, RGB LED indikátora a kapacitného dotykového tlačidla, ktoré je ukryté v prednom paneli. Schému zapojenia som vytvoril v programe KiCad a je dostupná na Github repozitári projektu. Ovládač nevyžaduje externé napájanie, keďže spotreba všetkých LED pri maximálnom zaťažení (biela farba, 100% jas) je približne 6*60mA = 360mA. 3D modely všetkých potrebných súčastí som vytvoril v programe Autodesk Fusion a sú dostupné na Githube projektu. Následne som všetky súčasti vytlačil na 3D tlačiarni.



Obrázok 1: výsledok zapojenia

Obrázok 2: hotový ovládač

Pri písaní Arduino kódu som sa stretol s nasledujúcim problémom: čítanie hodnôt z potenciometrov pomocou funkcie analogRead() nebolo stabilné, ak v danom momente svietili LED diódy. Zistil som, že tento problém priamo súvisel s danými LED. Tieto LED majú v sebe zabudovaný mikročip, ktorý komunikuje s Arduinom, a riadi jas jednotlivých diód vnútri pomocou metódy PWM (Pulse Width Modulation). Znamená to, že diódy sa veľmi rýchlo (s frekvenciou asi 400Hz) zapínajú a vypínajú, čo spôsobovalo problém s čítaním analógových hodnôt. Vyriešil som to pripájkovaním 330µF kondenzátora ku kladnému a zápornému pólu napájania LED.

Riadenie všetkých LED je možné s pomocou knižnice FastLED. Na komunikáciu s PC som vytvoril nasledujúci protokol:

- ?V – žiadosť o zaslanie hlasitostí všetkých kanálov (posiela PC -> Arduino)
- ?M – žiadosť o zaslanie stavu stlmenia (muted, not muted) všetkých kanálov (posiela PC -> Arduino)
- ?A – žiadosť o zaslanie stavu (active, inactive) všetkých kanálov (posiela Arduino -> PC)
- !V3:50 – príkaz: nastav hlasitosť kanála č.3 na 50%
- !V:10|20|30|40|50|60| – príkaz: nastav hlasitosť kanálov (1 - 6) na 10%, 20%, ...
- !M2:1 – príkaz: stlmiť kanál č.2 (posiela Arduino -> PC)

Program na Windows (Backend)

Hlavným cieľom v zimnom semestri bolo naučiť sa pracovať s knižnicou Win32 Core Audio API na ovládanie hlasitostí výstupných zariadení, ako aj jednotlivých aplikácii a programov bežiacich na PC so systémom Windows. Program, ktorý som vytvoril, beží na pozadí a čaká na komunikáciu cez sériový port s Arduinom a spracováva príkazy a požiadavky. Udržiava si zoznam výstupných zariadení spolu s tzv. „Audio Sessions“, čo sú vlastne zvukové výstupy jednotlivých aplikácii. Používateľovi umožňuje nastavenie COM portu, prenosovej rýchlosti, ale hlavne priradenie konkrétnych aplikácii k niektorému zo šiestich kanálov. (V jednom kanáli môže byť priradených viacero aplikácii.) Na interakciu s používateľom program pridáva ikonu na panel úloh, ktorá pri kliknutí pravým tlačidlom zobrazí kontextové menu. Položky v menu :

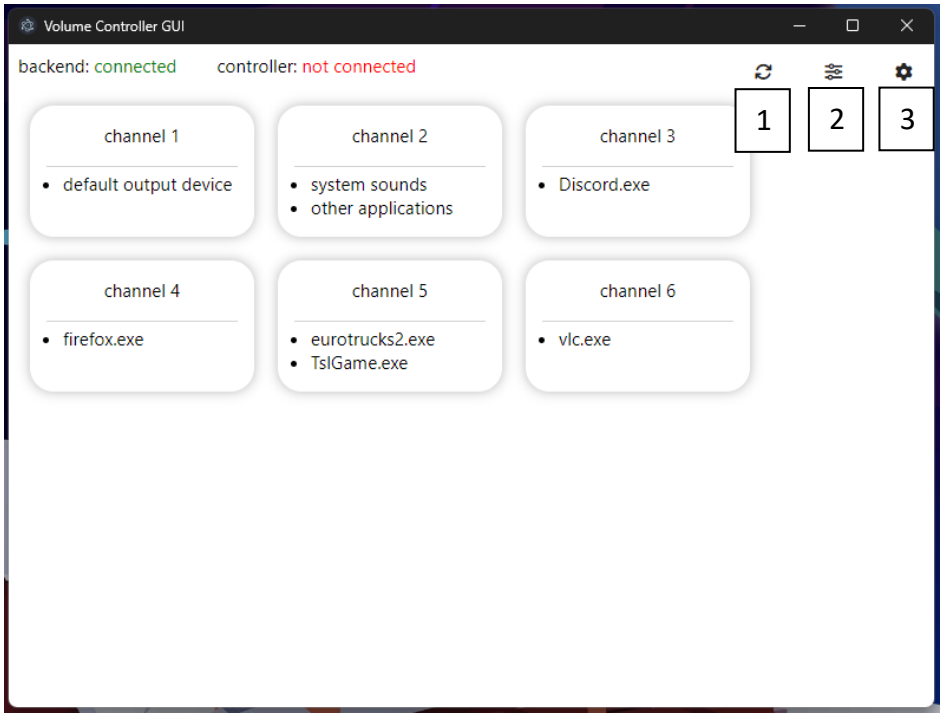
- „Open settings“ – otvorí okno, kde môže používateľ zmeniť nastavenia ako ktorý sériový port sa používa, alebo namapovať jednotlivé audio sessions ku konkrétnym kanálom,
- „Reconnect“, čiže zavrieť a opätovne otvoriť sériový port v prípade potreby,
- „Rescan audio sessions“ – opätovne zostaviť zoznam všetkých audio sessions a výstupných zariadení a namapovať ich k jednotlivým kanálom,
- „Debug info“ – zobrazí MessageBox,
- „Exit“ – ukončí program bežiaci na pozadí

Program tiež zachytáva rôzne eventy:

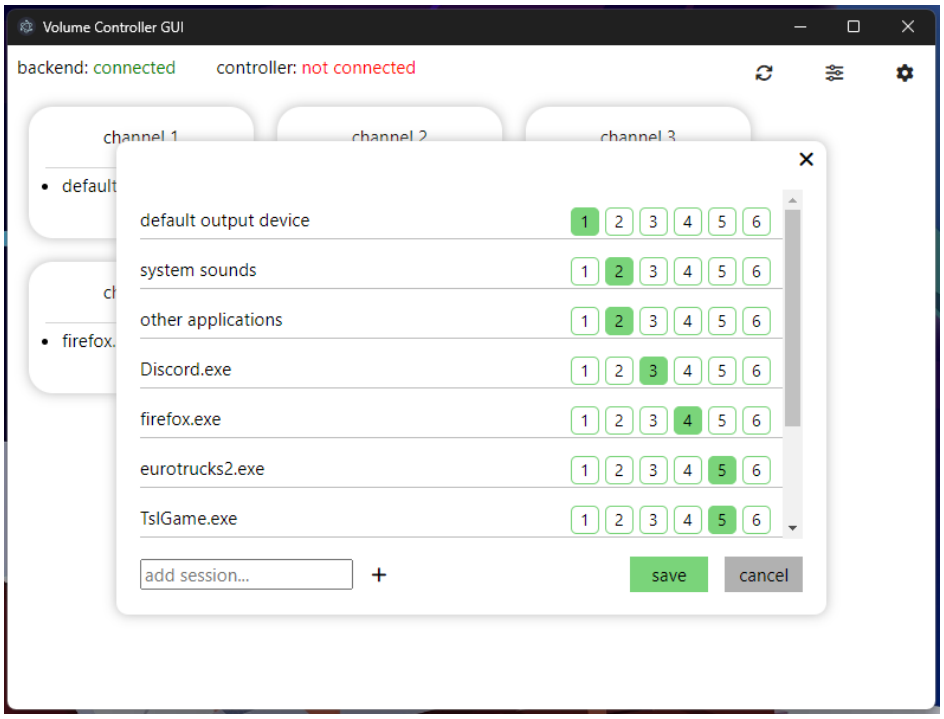
- Pripojenie a odpojenie COM zariadenia (na automatické otvorenie sériového portu)
- Vytvorenie novej Audio Session – ak používateľ spustí aplikáciu alebo program, ktorý vydáva zvuk, tak môj program túto session automaticky namapuje do správneho kanála (ak bol nejaký nastavený)
- Zánik nejakej Audio Session – program túto session odstráni zo všetkých priradených kanálov
- Zmena predvoleného výstupného zariadenia – vykoná sa operácia „Rescan Audio Sessions“ popísaná vyššie.

GUI

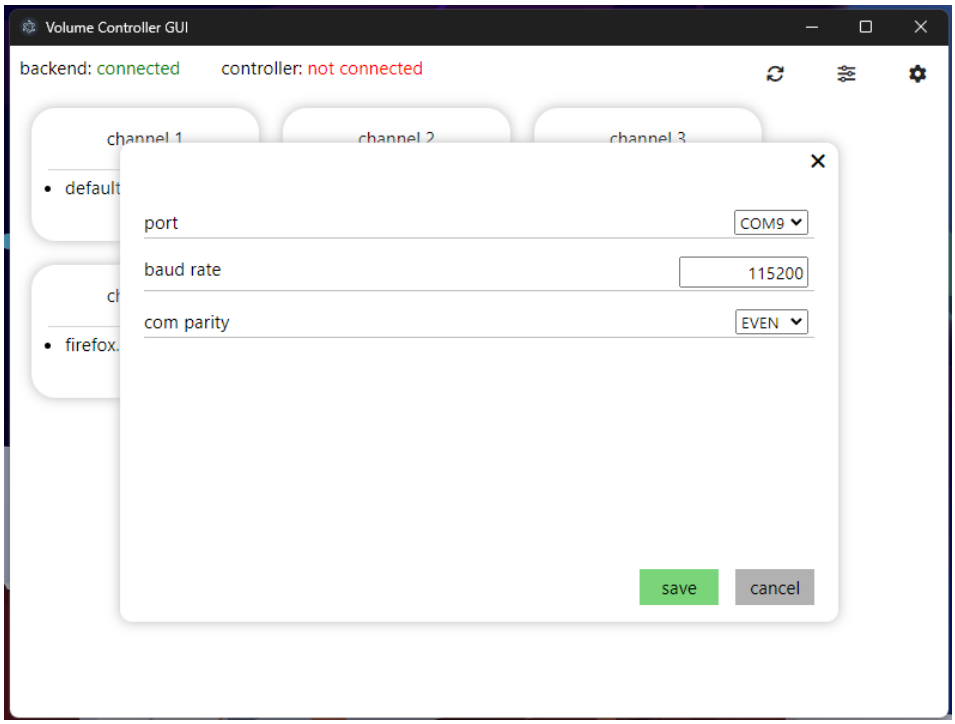
Ďalším cieľom v zimnom semestri bolo naučiť sa pracovať s platformou ElectronJS na vytváranie prenosných UI a vytvoriť grafické rozhranie, v ktorom používateľ bude môcť meniť rôzne nastavenia týkajúce sa COM portu, priradiť jednotlivé výstupné zariadenia a audio sessions na PC k jednému zo šiestich dostupných kanálov, ako aj pozrieť si prehľad o tom, čo je kam priradené a ktoré audio sessions sú práve aktívne. Komunikácia Electron aplikácie s windowsovým programom prebieha cez named pipe vo formáte JSON. GUI si môže od backendu vyžiadať obsah konfiguračného súboru, zoznam výstupných zariadení a audio sessions. Backendu potom posiela zmenené nastavenia. Tlačidlo označené číslom 1 vykoná „refresh“ dát – pošle backendu požiadavku na zaslanie vyššie vymenovaných dát. Tlačidlo 2 otvorí dialógové okno, kde používateľ priraďuje jednotlivé audio sessions a zariadenia ku kanálom. Tlačidlo 3 otvorí dialógové okno, v ktorom je možné meniť nastavenia týkajúce sa COM portu.



Obrázok 3: Electron aplikácia



Obrázok 4: priraďovanie sessions ku kanálom



Obrázok 5: nastavenia

Ciele v letnom semestri:

- Zistiť aké sú možnosti, čo sa týka systémových knižníc na ovládanie hlasitostí aplikácií na linuxových distribúciách
- Naučiť sa pracovať s danými knižnicami
- Naprogramovať softvér s rovnakou funkcionalitou na Linux (backend)
- Prispôsobiť UI zo zimného semestra, aby bol kompatibilný s Linuxovým backendom