FAKULTA INFORMAČNÍCH TECHNOLOGIÍ VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

ISA - Síťové aplikace a správa sítí – projekt
Discord bot

2020 Jan Hranický (xhrani02)

1 Úvod do problematiky

Zadáním tohoto projektu bylo vytvořit program, který bude zastávat roli bota na komunikační platformě Discord [1] . Po spuštění bot pomocí uživatelem dodaného autorizačního tokenu naslouchá na kanále #isa-bot a pomocí protokolu HTTP [2] a Discord API odpovídá na zprávy uživatelů následujícím formátem echo: [username] - [message]. Zprávy sebe sama a ostatních botů ignoruje.

1.1 API

API je zkratka pro Application Programming Interface. Česky přeloženo jako rohrání pro programovaní aplikací. API je sbírka procedůr, funkcí, tříd či protokolů nějáké knihovny, které může programátor využívat jako programové celky. Programátor je používá namísto toho aby je sám naprogramoval **APIâ…¥Wiki79:online**.

1.2 Discord API

Pro komunikaci s Discord serverem jsem v projektu využil Discord API, se kterým komunikuji přes protokol HTTPS. Discord api vrací data ve formátu JSON. V komunikaci s Discord api byly použity pouze požadavaky GET a POST

1.2.1 HTTPS

HTTPS je zkratka pro Hyptertext Transfer Protocol Secure. Spolu s protokolem HTTP využívá protokolu SSL nebo TLS pro šifrovanou komunikaci. Zajišťuje autentizaci, důvěrnost a integritu přenášených dat. Standartním portem pro protokol HTTPS je port 443.

1.2.2 GET

Požadavek GET se používá pro vyžádání specifikováného souboru. V projektu jsem použil následující GET požadavky:
GET/users/@me/guilds
GET/guilds/guild.id/channels
GET/channels/channel.id
GET/channels/channel.id/messages

1.2.3 POST

Na druhé straně spektra, požadavek POST se využívá k zaslání dat serveru. V projektu jsem pomocí požadavku POST posílal zprávy na Discord API. K tomu jsem využil konkrétní požadavek:
POST/channels/channel.id/messages

1.3 JSON

JSON [3] je zkratka pro JavaScript Object Notation, je to způsob přenosu dat, který je nazávislý na počítačové platformě. Data ve formátu JSON jsou organizována v polích nebo agregována v objektech. Příklad formátu JSON obdrženého přímo z Discord API :

```
{
    "id":"776752463986294785",
    "type":0,
    "content":"\"",
    "channel_id":"762106839054811176",
    "author":{
        "id":"487706666905894923",
        "username":"Emzak",
        "avatar":"a70859ecda1355dfd55bddcfd0194458",
        "discriminator":"6235",
        "public_flags":0
    },
    "attachments":[

    ],
    "embeds":[

    ],
    "mentions":[

    ],
    "mention_roles":[

    ],
    "pinned":false,
    "mention_everyone":false,
    "tts":false,
    "timestamp":"2020-11-13T10:16:58.777000+00:00",
    "edited_timestamp":null,
    "flags":0
}
```


2 Návrh a implementace

Projekt je rozdělen do čtyř modulů

- `argumentParser.cpp`
- `httpsClient.cpp`
- `jsonParser.cpp`
- `isabot.cpp`

Každý z modulů má také svůj hlavičkový soubor .hpp

2.1 argumentParser.cpp

Úkolem modulu `argumentParser.cpp` je pársovaní vstupních parametrů programu to se děje ve funkci `proccessArgume`
Funkce podle vstupních parametrů také nastavuje globální proměnné programu.

2.2 httpsClient.cpp

Modul `httpsClient` zajišťuje komunikaci programu s Discord API pomocí ssl socketu. Obsahuje následující funkce:

- `SendPacket` - funkce zašle pomocí socketu požadavek
- `initSSL` - pomocí této funkce se ustálí spojení skrze ssl socket. Spojení se při každém zaslaném požadavku vytváří a následně ukončuje.
- `sendRqAndGetResponse` - funkce zašle pomocí funkce `SendPacket` požadavek a následně přečte a vrátí jeho odpověď. Dále funkce kontroluje kód odpovědi požadavku. Při kódu 200 vrací odpoveď, při kódech 500 - internal server error a 421 - Too Many Requests se funkce volá rekurzivně dokud požadavek neprojde. V ostatních případech se volá funkce `logResponseCode`
- `logResponseCode` - funkce na stderr vypíše chybový kód a ukončí program

2.3 jsonParser.cpp

Modul `jsonParser` zpracovává odpoveď DiscordAPI formátu JSON. Navíc obsahuje několik dalších funkcí pro práci s datovým typem `string`. Pro zpracování JSONu jsou využity regulární výrazy, které pouze vytahují data z opovědi. Modul obsahuje tyto funkce pro zpracování formátu JSON:

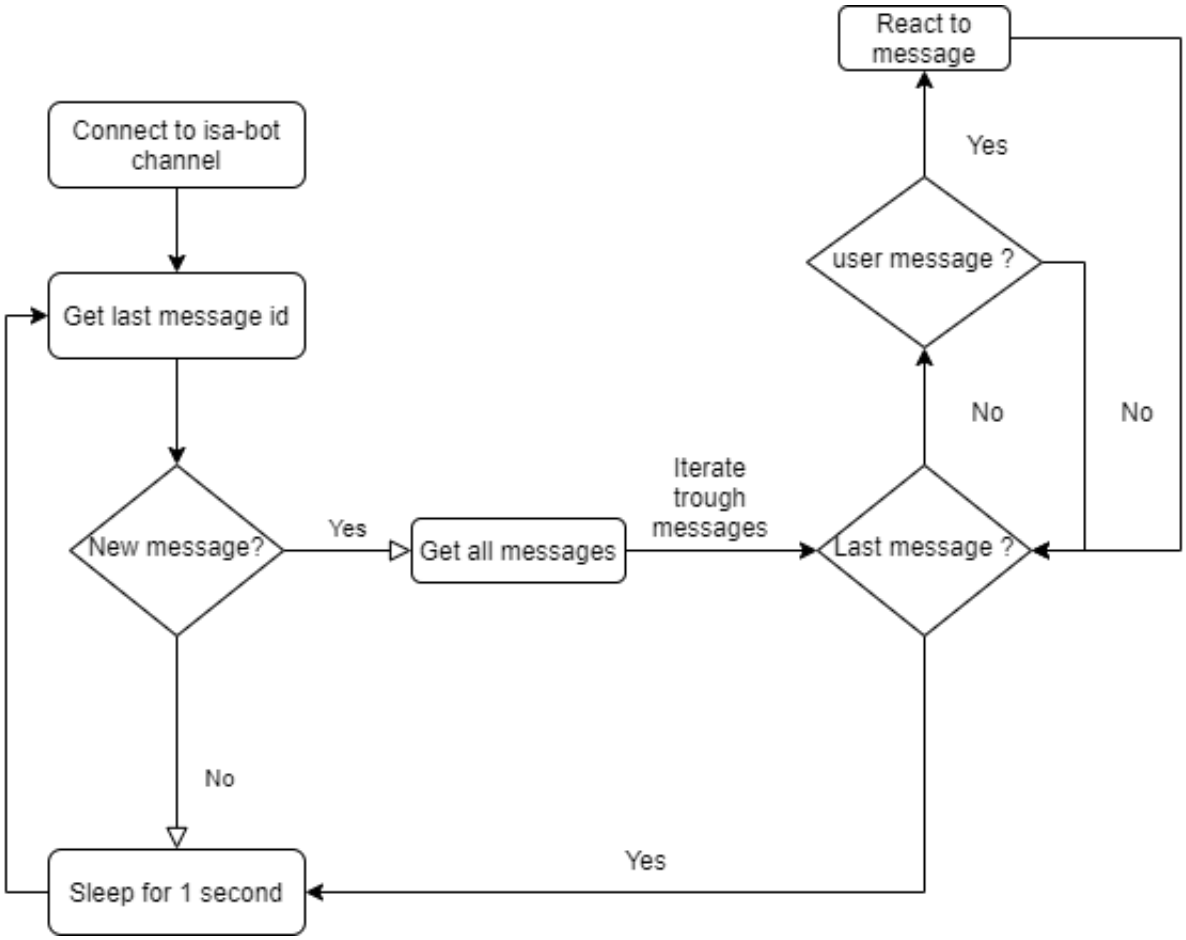
- `parseLastMessageId` - vrací id poslední zprávy kanálu isa-bot
- `parseGuildId` - vrací id serveru na který je isa-bot přídán
- `parseMessages` - vrací zpracované zprávy kanálu isa-bot v datové struktuře `std::map`
- `parseChannels` - vrací id kanálu isa-bot
- `executeRegex` - pomocná funkce, která extrahuje pattern z řetězce a vrací první nález
- `extractAllRegexPatterns` - pomocná funkce, která extrahuje pattern z řetězce a vrací všechny nálezy
- `constructParsedMsgMap` - pomocná funkce, která z zpracovaných zpráv vytvoří strukturu `std::map`

Navíc modul obsahuje tyto pomocné funkce pro práci s datovým typem `string`:

- `convertToString` - převádí řetězec jazyka C do datového typu `string` jazyka c++
- `splitString` - rozdělí `string` na dvě části za pomocí delimiteru
- `extractResponseCode` - za pomocí regulárních výrazů vrací kód http odpovědi
- `isBot` - vrací true v případě, že uživatelské jméno obsahuje podřetězec bot

2.4 isabot.cpp

Hlavní modul programu, obsahuje funkci `main()` a hlavní smyčku programu viz **??**.



Obrázek 1: Hlavní smyčka programu

3 Spuštění programu

Program se přelouží souborem Makefile pomocí příkazu Make. Samotný program se pouští následovně:

```
./isabot [-v] [-h] -t bot_token
-v - výpis zpráv, na které bot reagoval do konzole
-h - výpis nápovědy k programu
-t - bot_token pro komunikaci s Discord API
```

Odkazy

[1] *Discord Developer Portal — Documentation — Intro*, `https://discord.com/developers/docs/intro`, (Accessed on 11/17/2020).

[2] *RFC 8259 - The JavaScript Object Notation (JSON) Data Interchange Format*, `https://tools.ietf.org/html/rfc8259`, (Accessed on 11/17/2020).

[3] *RFC 8259 - The JavaScript Object Notation (JSON) Data Interchange Format*, `https://tools.ietf.org/html/rfc8259`, (Accessed on 11/17/2020).