Gymnázium, Praha 6, Arabská 14

Arabská 14, Praha 6, 160 00

Ročníková Práce

Předmět : Programování
Téma : Robot pro platformu Discord

Autor : Matěj Kratochvíl
Třída : 1.E
Školní rok : 2021/22
Vyučující : Mgr. Jan Lána
Třídní učitel : Mgr. Blanka Hniličková

Čestné prohlášení :

Prohlašuji, že jsem jediným autorem tohoto projektu, všechny citace jsou řádně označené a všechna použitá literatura a další zdroje jsou v práci uvedené. Tímto dle zákona 121/2000 Sb. (tzv. Autorský zákon) ve znění pozdějších předpisů uděluji bezúplatně škole Gymnázium, Praha 6, Arabská 14 oprávnění k výkonu práva na rozmnožování díla (§ 13) a práva na sdělování díla veřejnosti (§ 18) na dobu časově neomezenou a bez omezení územního rozsahu.

V Praze dne ……. ……..

Matěj Kratochvíl

Rád bych na tomto místě poděkoval panu profesorovi Mgr. Janu Lánovi, který dohlížel a kontroloval průběh ročníkové práce. Dále bych chtěl také poděkovat svojí matce Veronice Kratochvílové za další kontrolu textu a dokumentace. Děkuji jim za pomoc a podporu v průběhu mé práce.

Anotace

V rámci této ročníkové práce předkládám program v jazyku Java, který pomocí kódu umožňuje zpracování příkazů přijatých skrze komunikační platformu Discord. Cílem práce je vytvořit program, který by dokázal zpracovat příkazy a tím urychlil uživatelskou práci se serverem. K vytvoření programu byla využita veřejně dostupná Java Knihovna JDA (Java Discord API). Projekt obsahuje základní příkazy pro uživatelskou administraci vlastního serveru (odebrání přístupu nebo vyhození ze serveru) a příkazy pro odesílání obrázků. Dále také obsahuje funkce k odesílání zpráv při událostech připojení a odpojení ze serveru.

Annotation (English)

As part of this coursework, I present a Java program that allows the processing of commands received through the Discord communication platform. The aim of this work is to create a program that can process commands and thus speed up user´s work within their server. To create the program, the publicly available Java Library JDA (Java Discord API) was used. The project also includes basic commands for user server administration (removing access or kicking out of the server) and commands for sending images. The program also includes functions for sending messages at user joining and leaving guild event.

Úvod

Předmětem této ročníkové práce je program napsán v programovacím jazyce Java, který umožňuje zpracování uživatelských příkazů přijatých skrze platformu Discord.

Tento projekt jsem si vybral, protože s platformou Discord pracuji velmi často a často pracuji i s ostatními roboty vytvořenými někým jiným a vždy jsem si chtěl vytvořit i svého robota u kterého bych si jeho funkce mohl upravit podle sebe a nemusel bych být omezený cizím programem.

Pro realizaci projektu jsem využil programovací jazyk Java, vývojové prostředí Intellij Idea Community, který také využívám v průběhu hodin programování, a veřejně dostupnou Java knihovnu JDA.

1 Vývojové Prostředí a Platformy

Program je vytvořen v jazyce Java 18.0.1 pomocí vývojového prostředí IntelliJ IDEA Community Edition 2022.1 s využitím knihovny Java Discord API ve verzi 5.0.0alpha4

1.1 Java

“Java je objektově orientovaný programovací jazyk, který vyvinula firma Sun Microsystems a představila 23. května 1995.” [“WPJava”]

Uživatelskou verzi jazyku Java lze zdarma stáhnout na webových stránkách https://java.com/ a programátorskou na webových stránkách https://oracle.com/.

1.2 Java Discord API

České znění : *“JDA se snaží poskytovat, co nejlepší a nejrychlejší spojení Discord REST API a jeho Web-Socket eventů. Jedná o užitečný nástroj, který poskytuje funkce pro tvorbu discord bota v jazyce Java.”* [“DV8FromTheWorld”]

Knihovna je veřejně dostupná ke stažení na stránce GitHubu tvůrce https://github.com/DV8FromTheWorld/JDA.

V tomto projektu jsem z této knihovny využil metody listenerů pro příjem a zpracování zpráv, událostí s uživateli a odesílání zpráv ze strany robota.

1.3 IntelliJ IDEA

“IntelliJ IDEA je komerční vývojové prostředí pro programování v jazycích Java, Groovy a dalších.” [“WPIntellij”]

Vývojové prostředí IntelliJ IDEA Community lze zdarma získat na webových stránkách https://www.jetbrains.com/idea/.

1.4 Discord

Jedná se o komunikační platformu vyvinutou v roce 2015. Je na komunitní bázi, což znamená, že tvůrci poskytují pouze prostředí a funkce uživatelům, kteří si zde mohou vytvářet servery, roboty a vše ostatní podle svých představ.

Obsahuje funkce hlasových hovorů, které jsou založeny na technologii Voice Over Internet Protocol, neboli přímé spojení uživatelů přes jejich IP Adresy. Dále nabízí možnost textových chatů a video hovorů.

2 Instalace

2.1 Získání Tokenu pro Robota

Pro získání autentifikačního tokenu pro robota je zapotřebí založený a aktivovaný uživatelský účet na platformě Discordu, který je možné zdarma zaregistrovat na oficiálních stránkách https://discord.com/register.

Dalším krokem je založení samotné aplikace robota na stránkách Discordu pro tvůrce a developery https://discord.com/developers/applications, zde vyberete založení tlačítkem *New Application* na pravé horní části okna. Dále přejdeme k tvorbě, vyberte jméno vaší aplikace (toto jméno jde kdykoli změnit a neurčuje název robota). Stránka vás přesměruje na informace o aplikaci, zde z nabídky na levé straně vyberte kolonku *Bot*. Poté stiskněte tlačítko *Add Bot* a odsouhlaste založení. Následně uvidíte okno s kompletními informacemi o samotném robotovi.

Pro obdržení samotného Tokenu je nyní potřeba ho resetovat. Učinit tak můžete v horní části sekce informacích o roboti stiskněte tlačítko *Reset Token*. Nyní můžete vidět Token vašeho robota. Po uzavření této záložky bude pro znovu ukázání Tokenu potřeba ho znovu resetovat.

2.2 Přidání robota na server

K přidání robota na vlastní server nebo server, kde disponujete administrátorskými oprávněními, slouží opět stránka s informacemi o robotovi (https://discord.com/developers/applications).

Na stránce s informace z možností na levé straně vyberte druhou možnost *OAuth2*. Zde vyberte zaměření robota, pro tento případ vyberte pouze možnost *bot.* Dále se vám zobrazí další možnost, nyní s oprávněními pro robota, vyberte oprávnění *Administrator*. V aktuální chvíli vám stačí pouze zkopírovat odkaz ve spodní části okna a vložit ho do webového prohlížeče.

V okně pozvání robota na server vyberete váš cílový server pozvání a kliknout na *Pokračovat* a nadále *Autorizovat*. Tímto by váš robot měl být na vybraném serveru.

3 Program

src/Main/Main.java	● Připojení kódu k API Discordu (Token [60 místný kód]) ● Nastavení aktivity ● Připojení ostatních Java tříd ● Nastavení způsobu přijímání příkazů ● Konečné spuštění
src/Main/command/commands.java	● Nastavení předpony pro příkazy ● Konzolová zpráva po spuštění ● Způsob přijímání a vnímání příkazů
src/Main/listener/onJoinListener.java	● Odeslání textové zprávy do textového kanálu Discordu při připojení uživatele k serveru
src/Main/listener/onLeaveListener.java	● Odeslání textové zprávy do textového kanálu Discordu při odpojení uživatele ze serveru

3.1 Třída main

Třída main.java se zabývá prvotním spojením robota s API Discordu přes 60 místný kód (Token)

```
JDABuilder jda = JDABuilder.createDefault(“TOKEN”)
```

Nastavením statusu herní aktivity na profilu robota

```
jda.setActivity(Activity.watching("Made by Prach#9333"));
jda.setStatus(OnlineStatus.DO_NOT_DISTURB);
```

Spojením a spuštěním ostatních tříd programu

```
jda.addEventListeners(new commands(), new
onJoinListener(), new onLeaveListener());
```

Nastavením filtrů a způsobu přijimání uživatelů a zpráv

```
jda.setChunkingFilter(ChunkingFilter.ALL);
jda.setMemberCachePolicy(MemberCachePolicy.ALL);
jda.enableIntents(GatewayIntent.GUILD_MEMBERS);
```

a spuštěním kompletního programu

```
jda.build();
```

3.2 Třída commands

Třída commands.java se zabývá primárně nastavením všech příkazů robota a obsahuje největší část obsahu programu a nastavením předpony příkazů (prefixu)

3.2.1 Příkaz Ban

Příkaz *ban* slouží k permanentní blokaci uživatele ze serveru.

Používá se ve formátu *p!ban @[Zmínka uživatele k blokaci]*.
Robot dále čeká na zprávu v tomto formátu

```
if (args[0].equalsIgnoreCase(prefix + "ban"))
```

Dále si robot zkontroluje, jestli má odesílatel zprávy na daném serveru práva k manuální blokaci uživatelů,

```
if (event.getMember().hasPermission(Permission.BAN_MEMBERS))
```

pokud ano vytvoří tabulku s výpisem blokace, uživatele zablokuje a tabulku vyčistí pro další použití.

```
Member member = event.getGuild().getMember(event.getMessage()
.getMentionedUsers().get(0));
event.getGuild().ban(member, 0).queue();

EmbedBuilder ban = new EmbedBuilder();
ban.setColor(Color.GREEN);
ban.setTitle("Ban");
ban.setDescription("Výpis banu");
ban.addField("Autor Banu : ",
event.getMessage().getAuthor().getName(), true);
ban.addField("Ban : ", member.getUser().getName(), true);
ban.setFooter("Made by Prach#9333 <3");

event.getChannel().sendMessageEmbeds(ban.build()).queue();

ban.clear();
```

Pokud ne, vytvoří tabulku s `Ban ERROR` a textem `Nikdo nebyl zabanován, MISSING PERMISSIONS` a tabulku vyčistí.

```
} else {
Member member = event.getGuild().getMember(event.getMessage()
.getMentionedUsers().get(0));

   EmbedBuilder ban = new EmbedBuilder();
   ban.setColor(Color.RED);
   ban.setTitle("Ban ERROR");
   ban.setDescription("Nikdo nebyl zabanován, MISSING
PERMISSIONS");
   ban.addField("Autor Banu : ",
event.getMessage().getAuthor().getName(), true);
   ban.addField("Ban : ", member.getUser().getName(), true);
   ban.setFooter("Made by Prach#9333 <3");

   event.getChannel().sendMessageEmbeds(ban.build()).queue();

   ban.clear();
```

3.2.2 Příkaz Kick

Příkaz *kick* slouží k vyhození uživatele ze serveru

Používá se ve formátu *p!kick [Zmínka Uživatele k vyhození]*.
Robot dále čeká na zprávu v tomto formátu

```
if (args[0].equalsIgnoreCase(prefix + "kick"))
```

Dále si robot zkontroluje, jestli má odesílatel zprávy na daném serveru práva k manuálnímu vyhození uživatelů,

```
if (event.getMember().hasPermission(Permission.KICK_MEMBERS))
```

pokud ano vytvoří tabulku s výpisem vyhození, uživatele vyhodí a tabulku vyčistí pro další použití.

```
Member member = event.getGuild().getMember(event.getMessage()
.getMentionedUsers().get(0));
event.getGuild().kick(member).queue();

EmbedBuilder kick = new EmbedBuilder();
kick.setColor(Color.GREEN);
kick.setTitle("Kick");
kick.setDescription("Výpis vyhození");
kick.addField("Autor : ",
event.getMessage().getAuthor().getName(), true);
kick.addField("Kick : ", member.getUser().getName(), true);
kick.setFooter("Made by Prach#9333 <3");

event.getChannel().sendMessageEmbeds(kick.build()).queue();

kick.clear();
```

Pokud ne, vytvoří tabulku s `Kick ERROR` a textem `Nikdo nebyl vyhozen, MISSING PERMISSIONS` a tabulku vyčistí.

```
} else {
   Member member =
event.getGuild().getMember(event.getMessage().getMentionedUser
s().get(0));
   EmbedBuilder kick = new EmbedBuilder();
   kick.setColor(Color.RED);
   kick.setTitle("Kick ERROR");
   kick.setDescription("Nikdo nebyl vyhozen, MISSING
PERMISSIONS");
   kick.addField("Autor : ",
event.getMessage().getAuthor().getName(), true);
   kick.addField("Kick : ", member.getUser().getName(), true);
   kick.setFooter("Made by Prach#9333 <3");

   event.getChannel().sendMessageEmbeds(kick.build()).queue();

   kick.clear();
```

3.2.3 Příkaz kočka

Příkaz *kočka* slouží k odeslání náhodného obrázku kočky ze seznamu

Používá se ve formátu *p!kocka*
Robot dále čeká na zprávu v tomto formátu

```
if (args[0].equalsIgnoreCase(prefix + "kocka"))
```

Dále si robot vylosuje náhodné číslo v rozpětí velikosti seznamu

```
String[] kocek = {SEZNAM OBRÁZKŮ .GIF V PLATFORMĚ Tenor.com}

Random kocka = new Random();
int k = kocka.nextInt(kocek.length);
```

Po té na zprávu uživatele odpoví s odpovídajícím obrázkem

```
event.getMessage().reply(kocek[k]).queue();
```

3.2.4 Příkaz pes

Příkaz *pes* slouží k odeslání náhodného obrázku psa ze seznamu

Používá se ve formátu *p!pes*
Robot dále čeká na zprávu v tomto formátu

```
if (args[0].equalsIgnoreCase(prefix + "pes"))
```

Dále si robot vylosuje náhodné číslo v rozpětí velikosti seznamu

```
String[] psu = {SEZNAM OBRÁZKŮ .GIF V PLATFORMĚ Tenor.com}

Random pes = new Random();
int p = pes.nextInt(psu.length);
```

Po té na zprávu uživatele odpoví s odpovídajícím obrázkem

```
event.getMessage().reply(psu[p]).queue();
```

3.3 Třída onJoinListener

Třída onJoinListener.java slouží k odeslání zprávy do nastaveného kanálu při připojení uživatele k serveru.

Je založená na funkci `onGuildMemberJoin`, která čeká na událost připojení uživatele

```
public void onGuildMemberJoin(GuildMemberJoinEvent event)
```

Při této události odešle textovou zprávu *Nazdar* se zmínkou daného uživatele do textového kanálu nastaveného podle Identifikačního čísla kanálu

```
event.getGuild().getTextChannelById(664847658585686044L)
.sendMessage
        ("Nazdar " +
event.getMember().getUser().getAsMention()).queue();
```

3.4 Třída onLeaveListener

Třída onLeaveListener.java slouží k odeslání zprávy do nastaveného kanálu při odpojení uživatele ze serveru.

Je založená na funkci `onGuildMemberRemove`, která čeká na událost odebrání/odpojení uživatele

```
public void onGuildMemberRemove(GuildMemberRemoveEvent event)
```

Při této události odešle textovou zprávu *Pápá* se zmínkou daného uživatele do textového kanálu nastaveného podle Identifikačního čísla kanálu

```
event.getGuild().getTextChannelById(664847658585686044L)
.sendMessage
        ("Pápá " +
event.getMember().getUser().getAsMention()).queue();
```

4 Závěr

V rámci tohoto ročníkového projektu jsem naprogramoval robota pro platformu Discord, všechny uvedené a naprogramované funkce jsou plně funkční. Původně jsem uvažoval i o funkci přehrávání muziky do hlasového kanálu, ale pro to by muselo dojít k implementaci další knihovny a API, proto jsem se do této funkce nepouštěl.

I přes to byl pro mne projekt přínosný a obohatil moje zkušenosti s touto knihovnou a také mi přinesl možnost robota dále rozvíjet a používat.

5 Seznam zdrojů

4.1 Online Zdroje

["DV8FromTheWorld"]

"DV8FromTheWorld/JDA: Java wrapper for the popular chat & VOIP service: Discord https://discord.com." *GitHub*, 18 Březen 2022, https://github.com/DV8FromTheWorld/JDA. Accessed 1 May 2022.

["WPIntelliJ"]

"IntelliJ IDEA – Wikipedie." *Wikipedie*, 27 Prosinec 2021, https://cs.wikipedia.org/wiki/IntelliJ_IDEA. Accessed 1 May 2022.

["WPJava"]

"Java (programovací jazyk) – Wikipedie." *Wikipedie*, 2 1 2022, https://cs.wikipedia.org/wiki/Java_(programovac%C3%AD_jazyk). Accessed 1 May 2022.

4.2 Seznam Zdrojů Softwaru

https://oracle.com/
https://java.com/
https://www.jetbrains.com/idea/
https://github.com/DV8FromTheWorld/JDA
https://tenor.com