

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.

Anotace	4
Annotation (English)	4
Úvod	6
1 Vývojové Prostředí a Platformy	7
1.1 Java	7
1.2 Java Discord API	7
1.3 IntelliJ IDEA	7
1.4 Discord	7
2 Program	8
2.1 Třída main	9
2.2 Třída commands	10
2.2.1 Příkaz Ban	10
2.2.2 Příkaz Kick	12
2.2.3 Příkaz kočka	14
2.2.4 Příkaz pes	15
2.3 Třída onJoinListener	16
2.4 Třída onLeaveListener	17
3 Závěr	18
4 Seznam zdrojů	19
4.1 Online Zdroje	19
4.2 Seznam Zdrojů Softwaru	20

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>.

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

“Discord je komunikační platforma pro komunity. Je alternativou Skypu, TeamSpeaku, ICQ a ostatních služeb. Discord nabízí podporu botů, správu práv a jiných záležitostí. Discord lze používat na systémech Windows, MacOS, Linux a ve webovém prohlížeči.” [“WPDC”]

2 Program

src/Main/Main.java	<ul style="list-style-type: none">• 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	<ul style="list-style-type: none">• Nastavení předpony pro příkazy• Konzolová zpráva po spuštění• Způsob přijímání a vnímání příkazů
src/Main/listener/onJoinListener.java	<ul style="list-style-type: none">• Odeslání textové zprávy do textového kanálu Discordu při připojení uživatele k serveru
src/Main/listener/onLeaveListener.java	<ul style="list-style-type: none">• Odeslání textové zprávy do textového kanálu Discordu při odpojení uživatele ze serveru

2.1 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 přijímání uživatelů a zpráv

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

a spuštěním kompletního programu

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

2.2 Třída commands

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

2.2.1 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
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();
```

2.2.2 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();
```

2.2.3 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();
```

2.2.4 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();
```

2.3 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();
```


2.4 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();
```

3 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.

4 Seznam zdrojů

4.1 Online Zdroje

[“WPDC”]

“Discord (software) – Wikipedie.” *Wikipedie*, 12 Březen 2022,

[https://cs.wikipedia.org/wiki/Discord_\(software\)](https://cs.wikipedia.org/wiki/Discord_(software)).

Accessed 1 May 2022.

[“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_\(programovac%C3%AD_jazyk\)](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>